

BASIC PLUS

80 ROUTINES SUR MO5 ET TO7/70

MICHEL MARTIN

BASIC PLUS

80 ROUTINES SUR

MO5 ET TO7/70

Autres ouvrages relatifs aux MO5 et TO7/70

— La découverte des MO5 et TO7/70 - Maurice Charbit et Dominique Schraen
— Exercices en BASIC pour MO5 et TO7/70 - Maurice Charbit et Dominique Schraen
— MO5 et TO7/70 pour tous - Jacques Boisgontier
— 102 programmes pour MO5 et TO7/70 - Jacques Deconchat
— Super Jeux MO5 et TO7/70 - Jean-François Sehan
— MO5 et TO7/70 en famille - Jean-François Sehan
— MO5 et TO7/70 pour tout petits - Daniel Nielsen
— MO5 et TO7/70 à l'école - Daniel Nielsen
— MO5 et TO7/70 pour réussir en CM1 - Daniel Nielsen
— Destination Collège - Daniel Nielsen
— Profs assistance - Daniel Nielsen

A paraître :

— Supergénérateur de caractères - Jean-François Sehan
— Français et Thomson en sixième - Jacques Deconchat et Gisèle Sergeant
— Maths et Thomson en cinquième - Jacques Deconchat
— MO5 et TO7/70 pour réussir en CE2 - Daniel Nielsen
— Logo à l'école - Daniel Nielsen et Gérard Bossuet

Pour tout problème rencontré dans les ouvrages P.S.I.
vous pouvez nous contacter au numéro ci-dessous :

Numéro Vert/Appel Gratuit en France
05 21 22 01 (Composer tous les chiffres, même en région parisienne)

ISBN 2-86595-264-9

MICHEL MARTIN



BASIC PLUS

80 ROUTINES SUR

MO5 ET TO7/70



ÉDITIONS DU P.S.I.
1985

Sommaire

CHAPITRE 2 - AMÉLIORATIONS DES BASIC MO5 ET TO7/70 31

Avertissement

Ce livre s'adresse aux possesseurs de MO5 et TO7/70 qui ont déjà pratiqué le BASIC et qui veulent aller plus loin.

L'ouvrage propose un ensemble de 80 programmes utilitaires qui vous permettront de manipuler simplement le générateur sonore, le graphisme haute résolution, ou d'ajouter des fonctions au BASIC Thomson.

L'ouvrage se compose de six parties :

- les fonctions évoluées des MO5 et TO7/70 ;
- améliorations des BASIC MO5 et TO7/70 ;
- utilitaires d'écran ;
- le générateur sonore ;
- les modes graphiques haute résolution ;
- animation graphique.

□ **"Les fonctions évoluées des MO5 et TO7/70"** donne un ensemble d'outils pour mieux utiliser votre ordinateur.

□ **"Améliorations des BASIC MO5 et TO7/70"** permet d'ajouter des instructions au BASIC de cet ordinateur, comme répétition de chaîne, tracé d'un cercle, opérations sur les matrices, etc.

□ **"Utilitaires d'écran"** donne des outils pour simplifier l'affichage et la saisie à l'écran.

□ **"Le générateur sonore"** détaille la structure du générateur sonore, permet de programmer des morceaux de musique et simule divers bruits.

□ **"Les modes graphiques haute résolution"** détaille la structure de l'écran graphique, montre son utilisation et permet d'augmenter le nombre de couleurs possibles sur l'écran.

- **"Animation graphique"** définit le concept de programmation de caractères graphiques et montre son utilisation à travers des programmes très progressifs. Enfin, six programmes Assembleur permettent de sauvegarder et de restituer des images graphiques monochromes ou couleurs.

- Un **avant-propos** donne une méthode pour faciliter l'écriture et la lisibilité des programmes BASIC.

BASIC : partir sur de bonnes bases

LES ORIGINES DU BASIC

Le langage BASIC a été créé en 1964 aux États-Unis. La signification du sigle BASIC (Beginners All Purpose Symbolic Instruction Code, ou encore Codage d'instructions à usage général pour débutants) nous montre que ses auteurs le désignaient avant tout comme un langage simple, utilisable par les débutants.

Aujourd'hui, le BASIC a bien évolué et s'impose aussi bien sur la plupart des micro-ordinateurs familiaux que dans certaines applications professionnelles. Tout en gardant sa simplicité d'emploi, il a su devenir puissant et souple.

Le langage BASIC se compose d'un certain nombre de mots clés ou instructions. Deux machines différentes utilisant le langage BASIC n'auront pas exactement le même vocabulaire. En effet, certaines instructions varient d'un ordinateur à l'autre, en raison des possibilités particulières de chaque appareil. Ainsi, un ordinateur qui ne possède pas de générateur sonore n'a rien à faire de commandes de génération de son. Ou encore, un ordinateur ne travaillant qu'avec deux couleurs n'a rien à faire d'instructions de manipulation de couleurs.

Malgré cela, la plupart des mots clés du BASIC sont indépendants de l'ordinateur utilisé, de sa marque et de son pays de fabrication. On peut donc parler de l'universalité du BASIC, en sachant très bien que ce terme n'est pas entièrement vérifié...

Le BASIC ainsi défini, il peut sembler facile d'écrire un programme en mettant bout à bout un ensemble de mots clés.

Les paragraphes qui suivent montrent qu'il n'en est rien, et que l'écriture d'un programme BASIC demande méthode et réflexion.

MIEUX PROGRAMMER EN DÉFINISSANT SES OBJECTIFS

Considérons un programmeur débutant qui a l'intention d'écrire un programme de sa conception. Très souvent, il s'y prendra de la façon suivante : il s'assiéra devant la machine et commencera à taper ce qu'il croit être un programme, sans avoir pris le temps d'y réfléchir.

Deux cas peuvent se présenter :

- le programme à écrire est court et/ou simple ;
- le programme à écrire est long et/ou complexe.

Dans le premier cas, et avec un peu d'expérience, le programmeur pourra arriver à ses fins, après quelques tentatives infructueuses.

Dans le second cas, le programmeur peinera pour écrire un programme sans structure et très difficilement compréhensible pour lui-même ou pour d'autres programmeurs. Il pourra même ne plus comprendre ce qu'il a écrit, et arrêter son travail, découragé.

Heureusement, il existe une méthode pour permettre aux programmeurs, débutants ou expérimentés, d'écrire des programmes clairs et facilement compréhensibles, même si la tâche qu'ils réalisent est longue et complexe.

La clé de cette méthode réside dans l'"analyse du problème" que l'on se propose de résoudre par l'écriture d'un programme. Cette analyse débouche sur une définition des objectifs.

La plupart des programmes peuvent être mis sous la forme suivante :

ENTREE → TRAITEMENT → SORTIE

La sortie du programme est l'objectif à atteindre, le résultat à calculer ou à afficher.

L'entrée du programme représente les données nécessaires à la réalisation des sorties.

Le traitement représente l'ensemble des opérations à faire subir aux entrées pour obtenir les sorties désirées.

La définition des objectifs consistera à classer et décrire l'ensemble des entrées/sorties et le traitement. Plus cette description sera détaillée, plus il sera ensuite facile d'écrire les instructions correspondantes en langage BASIC.

Par exemple, pour un programme de jeu :

- les entrées peuvent être les différentes commandes de déplacement et de tir entrées au clavier ;
- les sorties peuvent être le déplacement de divers objets sur l'écran ;
- le traitement peut être la définition graphique des objets ; l'interprétation des commandes entrées par le joueur, le déplacement effectif des objets.

Arrivé à ce point, deux choses très importantes restent à faire : organiser ses objectifs en définissant une structure et hiérarchiser les fonctions du programme.

MIEUX PROGRAMMER EN ORGANISANT SES OBJECTIFS : CONCEPT DE STRUCTURE

Les entrées/sorties et le traitement étant définis, il est nécessaire de donner une structure au traitement. Autrement dit, il faut organiser la tâche plus ou moins complexe que constitue le traitement en un ensemble de sous-tâches plus simples, chaque sous-tâche étant définie par un traitement spécifique.

Prenons l'exemple suivant : le programme à réaliser a pour objectif le tracé d'une courbe représentée par une équation quelconque de la forme Y = f (x).

L'analyse des objectifs nous montre que les entrées sont :

- domaine d'étude ;
- équation de la courbe.

La sortie est : affichage du min et du max sur la courbe.

Le traitement est : résolution et affichage de l'équation entre les bornes du domaine d'étude choisi.

Si nous poursuivons l'analyse, le traitement peut se décomposer en :

- recherche de l'échelle en Y (pour faire apparaître la courbe sur tout l'écran),
- recherche du pas du tracé (distance entre deux points de l'axe Ox),
- tracé d'un point aux coordonnées X, Y.

Le traitement se décompose ici en trois sous-tâches élémentaires.

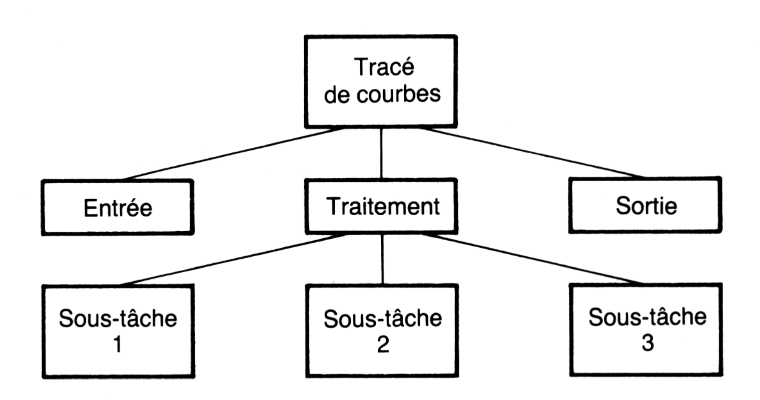
L'écriture d'un module ou procédure pour chaque sous-tâche constitue la structure du traitement.

Arrivé à ce niveau, il ne reste plus qu'une étape avant le codage puis l'écriture du programme sur la machine.

MIEUX PROGRAMMER EN DÉFINISSANT DES NIVEAUX : CONCEPT DE HIÉRARCHIE

Nous avons vu qu'il était nécessaire de diviser la tâche traitement en sous-tâches plus simples.

L'exemple précédent peut se schématiser comme suit :



Cette représentation ou "graphe hiérarchique" suffit à elle seule pour définir la structure du programme.

Pour cet exemple précis, la programmation BASIC sera la suivante :

```
10 REM Trace de courbes
20 :
30   GOSUB 1000 'Entree
40   GOSUB 2000 'Traitement
50   GOSUB 3000 'Sortie
60 :
70 END
80 REM ----------------------------
1000 REM Entree
1010 :
1020 REM Saisie de l'equation et du
1030 REM domaine de definition
1040 :
1050 RETURN
1060 REM ---------------------------
2000 REM Traitement
2010 :
2020 GOSUB 10000 'Echelle Ox
2030 GOSUB 11000 'Pas Ox
2040 FOR I=1 TO NBPTS STEP PAS
2050   GOSUB 12000 'Trace d'un point
2060 NEXT I
2070 :
2080 RETURN
2090 REM ---------------------------
3000 REM Sortie
3010 :
3020 REM Affichage des Min et Max
3030 :
3040 RETURN
3050 REM ---------------------------
10000 REM Echelle Oy
10010 :
10020 REM Recherche de l'echelle
10030 :
10040 RETURN
10050 REM --------------------------
11000 REM Recherche du pas Ox
11010 :
11020 REM Calcul du pas
11030 :
11040 RETURN
11050 REM --------------------------
12000 REM Affichage d'un point
12010 :
12020 REM Point aux coordonnees X,Y
12030 :
12040 RETURN
```

Cet exemple simple fait appel à une structure à trois niveaux hiérarchiques. Des programmes plus complexes peuvent demander quatre et même cinq niveaux hiérarchiques. Les sous-tâches se décomposent alors en blocs plus élémentaires qui se décomposent eux-mêmes en d'autres blocs plus élémentaires encore.

VERS UNE PROGRAMMATION EFFICACE

La dernière étape constitue l'écriture du programme. Les objectifs étant définis, le traitement structuré et hiérarchisé, vous pouvez probablement gagner du temps en tapant directement votre programme au clavier, plutôt que de l'écrire sur une feuille puis le recopier.

Deux derniers conseils :

- Affectez à chaque niveau hiérarchique un groupe de N^{os} de lignes.
  Ainsi, écrivez par exemple :
  le niveau 1 entre les lignes 10 et 100 ;
  le niveau 2 entre les lignes 100 et 1000 ;
  le niveau 3 entre les lignes 1000 et 10000 ;
  etc.
- Documentez votre programme en insérant un nombre non négligeable de remarques (REM) concernant les entrées, sorties et traitement de chaque module.

Ces tâches peuvent vous paraître fastidieuses, mais, si vous les suivez méthodiquement, vous vous féliciterez de la facilité avec laquelle vous pourrez insérer une modification six mois après la création du programme...

Chapitre 1 — Les fonctions évoluées des MO5 et TO7/70

Ce chapitre donne la manière de se servir au mieux des fonctions BASIC des MO5 et TO7/70, ou de leurs routines internes.

Les fonctions simples, communes à la plupart des BASIC, ne sont pas reprises ici. L'étude porte sur cinq sujets :

- les fonctions logiques ;
- la fonction DEF FN ;
- le classement de nombres et de chaînes ;
- le chaînage de deux programmes avec passage d'arguments ;
- les routines Assembleur utiles.

LES FONCTIONS LOGIQUES

Fonctions des MO5 et TO7/70

AND

A	B	A AND B
0	0	0
0	1	0
1	1	1
1	0	0

Utilisation de AND

IF A=4 AND C=5 THEN PRINT

Une ligne sera sautée si A=4 et C=5.

OR

A	B	A OR B
0	0	0
0	1	1
1	1	1
1	0	1

Utilisation de OR

IF A=4 OR C=5 THEN 20

Le débranchement à la ligne 20 se fera si A=4 ou C=5.

NOT

A	NOT A
0	1
1	0

Utilisation de NOT

IF NOT A=B THEN PRINT

Si A=B n'est pas vrai, autrement dit si A<>B, alors une ligne est sautée.

XOR

A	B	A XOR B
0	0	0
0	1	1
1	1	0
1	0	1

La table de vérité de XOR est identique à celle du OU (OR), à ceci près que le cas A vrai et B vrai donne un résultat A XOR B faux.

Utilisation

La multiplication de deux nombres de signe contraire donne un nombre négatif, la multiplication de deux nombres de même signe donne un nombre positif. La fonction XOR peut être utilisée pour déterminer le signe du produit de deux nombres.

Soit A$ le signe de A, B$ le signe de B et P$ le signe de A*B. Alors,

IF A$="−" XOR B$="−" THEN P$="−"

permet d'affecter un signe négatif au produit si l'un des deux nombres est négatif.

IMP

A	B	A IMP B
0	0	1
0	1	1
1	1	1
1	0	0

Seul le cas A vrai et B faux donne A IMP B faux.

Utilisation

Soit une serrure, si vous possédez la clé (CLE=1) et si la serrure est fermée (FF=0), alors vous pouvez ouvrir la porte :

IF NOT ((CLE=1) IMP (FE=1)) THEN GOSUB OUVRIR

EQV

A	B	A EQV B
0	0	1
0	1	0
1	1	1
1	0	0

La table de vérité de EQV est obtenue en complémentant la table de vérité de XOR.

Utilisation

Pour reprendre l'exemple proposé dans XOR,

IF A$="−" EQV B$="−" THEN P$="+"

permet d'affecter un signe positif au produit si les deux nombres sont de même signe.

Remarque : les fonctions **AND, OR, NOT, XOR, IMP** et **EQV** s'étendent également aux nombres entiers compris entre −32768 et 32767. Le résultat est alors égal à la fonction appliquée à chaque bit des deux nombres impliqués.

La notation employée par les micro-ordinateurs MO5 et TO7/70 est inversée. Ainsi la condition vraie sera représentée par "−1" et la condition "faux" par "0".

D'autres fonctions logiques sont parfois utilisées sur des machines différentes. Voici les deux plus courantes et les programmes qui permettent de les simuler en logique booléenne (Sortie à 1 ou à 0).

Fonction NOR (Non OU)

A	B	A NOR B
0	0	1
0	1	0
1	1	0
1	0	0

→ Si A et B sont faux, alors A NOR B est vrai.

Utilisation

Soit un jeu dans lequel un avion à hélice est animé. Si l'hélice ne tourne pas et s'il n'y a plus de carburant, alors l'avion s'écrase.

Si HELICE=1 indique que l'hélice tourne et CA=1 indique qu'il reste du carburant, alors IF HELICE=1 NOR CA=1 THEN GOSUB ECRASEMENT lancera la procédure d'écrasement.

Le programme

A et B contiennent les variables à tester.
X contient A NOR B à la sortie du programme.

```
1 REM  **************************************
2 REM  *                                    *
3 REM  *      SIMULATION DE LA FONCTION     *
4 REM  *      LOGIQUE 'NOR'                 *
5 REM  *                                    *
6 REM  **************************************
7 REM  *                                    *
8 REM  *  Entree : Variables A et B         *
9 REM  *  Sortie : Resultat dans X          *
10 REM *                                    *
11 REM **************************************
12 :
20 CLS:INPUT"Premier parametre ";A
30 INPUT"Deuxieme parametre ";B
40 GOSUB 100 ' A NOR B
50 PRINT:PRINT A"NOR"B"="X
60 END
70 REM ------------------------------------
100 REM Calcul de la fonction NOR
101 :
110 X=0
120 IF A=0 AND B=0 THEN X=1
130 :
140 RETURN
```

Analyse du programme

Ligne 110 : Sortie par défaut.
Ligne 120 : Calcul de la fonction NOR.

Fonction NAND (Non ET)

A	B	A NAND B
0	0	1
0	1	1
1	1	0
1	0	1

Si A vrai et B vrai, alors A NAND B faux ; tous les autres cas donnent A NAND B vrai.

Utilisation

Le même exemple que précédemment peut être repris :

IF HELICE=1 NAND CA=1 THEN GOSUB DEFAILLANCE

En effet, si l'hélice ne tourne plus (HELICE = 1 faux), ou s'il n'y a plus de carburant (CA=1 faux), alors, la procédure DEFAILLANCE est lancée.

Le programme

A et B contiennent les variables à tester.
X contient A NAND B à la sortie du programme.

```
1 REM  ************************************
2 REM  *                                  *
3 REM  *     SIMULATION DE LA FONCTION    *
4 REM  *     LOGIQUE 'NAND'               *
5 REM  *                                  *
6 REM  ************************************
7 REM  *                                  *
8 REM  * Entree : Variables dans A et B   *
9 REM  * Sortie : Resultat dans X         *
10 REM *                                  *
11 REM ************************************
12 :
20 CLS:INPUT"Premier Parametre ";A
30 INPUT"Deuxieme Parametre ";B
40 GOSUB 100 ' A NAND B
50 PRINT:PRINT A"NAND"B"="X
60 END
70 REM -------------------------------------
100 REM Calcul de la fonction NAND
101 :
110 X=1
120 IF A=1 AND B=1 THEN X=0
130 :
140 RETURN
```

Analyse du programme

Ligne 110 : Sortie par défaut.
Ligne 120 : Calcul de la fonction NAND.

Émulateur de fonctions logiques

Pour terminer ce chapitre sur les fonctions logiques, voici un programme mettant en œuvre une fonction booléenne aussi complexe que vous le désirez, constituée des opérateurs booléens des MO5 et TO7/70.

Donnez le nombre de variables de la fonction à définir, puis entrez en 175 une fonction booléenne appelée A.

Exécutez le programme à partir de la ligne 50. La table de vérité de la fonction booléenne est alors affichée.

Remarque : les variables booléennes de la fonction A sont appelées V(i) où i varie de 1 à NV (nombre de variables).

Programme

```
1 REM  ***************************************
2 REM  *                                     *
3 REM  *            MINI   EMULATEUR         *
4 REM  *         DE  FONCTIONS  LOGIQUES     *
5 REM  *                                     *
6 REM  ***************************************
7 REM  *                                     *
8 REM  *  Entree : Fonction logique          *
9 REM  *           NV=Nombre de variables    *
10 REM *  Sortie : Table de verite           *
11 REM *                                     *
12 REM ***************************************
13 :
20 CLS:PRINT"Definissez votre fonction booleen
ne"
30 PRINT"en 175 puis tapez 'RUN 50'"
40 STOP
50 REM -----------------------------------
60  PRINT:INPUT"Nombre de variables ";NV
70  CLS
80  FOR P=0 TO 2^NV-1
90    K1=P
100   FOR I=NV-1 TO 0 STEP -1
110     K2=K1-2^I
120     IF K2<0 THEN V(NV-1-I)=0
130     IF K2>=0 THEN V(NV-1-I)=1:K1=K2
140   NEXT I
150   FOR I=0 TO NV-1
160     PRINT V(I);
170   NEXT I
175   A=(V(0) AND V(1)) OR V(2)
180   PRINT"!"A
190 NEXT P
```

Analyse du programme

Lignes 20 à 40	: Définition.
Ligne 60	: Saisie du nombre de variables.
Ligne 175	: Fonction booléenne étudiée.
Ligne 180	: Résultat de la fonction.

LES FONCTIONS RÉSERVÉES A L'UTILISATEUR : DEF FN

La fonction BASIC **DEF FN** permet de définir simplement une fonction mathématique qui sera utilisée dans la suite du programme.

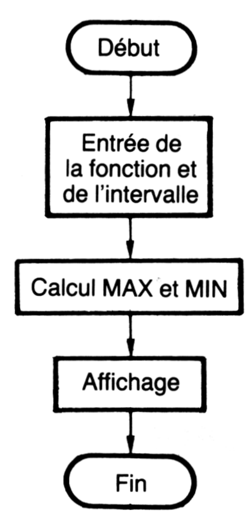
Le BASIC MO5 ne possède pas DEF FN, mais cette fonction peut facilement être simulée.

DEF FN va permettre à un utilisateur non programmeur de définir la fonction qui l'intéresse.

Exemple d'utilisation

Soit une fonction mathématique. Nous voulons connaître son maximum et son minimum dans un intervalle donné.

Ce problème se symbolise comme suit :



Programme

```
1000 REM ************************************
*****
1010 REM * Recherche du minimum et du maximum
    *
1020 REM * d'une fonction sur un intervalle d
onne*
1030 REM ************************************
*****
1040 :
1050 CLS:PRINT"Tapez '1190 A=' suivi de la"
1060 PRINT"fonction a etudier."
```

```
1070 PRINT:PRINT"Tapez ensuite 'RUN 1100'"
1080 STOP
1090 :
1100 REM Saisie de l'intervalle d'etude
1110 PRINT:PRINT"Entrez l'intervalle d'etude
:"
1120 PRINT:INPUT"Minimum ";MI
1130 PRINT:INPUT"Maximum ";MA
1140 :
1170 REM Calcul des Minimum et Maximum
1175 :
1176 U=-1E+15:V=1E+15
1180 FOR X=MI TO MA STEP (MA-MI)/100
1190   A=SIN(X)
1200   IF A>U THEN U=A 'Maximum
1210   IF A<V THEN V=A 'Minimum
1220 NEXT X
1230 :
1240 REM Affichage des resultats
1245 :
1250 PRINT:PRINT"Le maximum est :";U
1260 PRINT:PRINT"Le minimum est :";V
1270 END
```

Analyse du programme

Lignes 1050 à 1080	: Présentation.
Lignes 1100 à 1130	: Saisie de l'intervalle d'étude.
Ligne 1190	: Définition de la fonction.
Lignes 1170 à 1220	: Calcul des min et max.
Lignes 1240 à 1270	: Affichage des min et max.

CLASSEMENT PAR ORDRE CROISSANT ET DÉCROISSANT

Quel informaticien n'a jamais eu des données à classer par ordre croissant ou décroissant ? Sans doute n'en existe-t-il pas.

Le langage BASIC fait la différence entre caractères et nombres, c'est ainsi que deux programmes sont présentés :

- un pour le classement des chaînes de caractères,
- un pour le classement des nombres.

Il existe un nombre impressionnant de méthodes de classement par ordre croissant et décroissant. Celle qui a été retenue ici offre deux avantages : sa rapidité et sa simplicité.

Méthode employée pour le classement

Examinons le cheminement du programme de classement par ordre croissant. Le classement par ordre décroissant est très similaire.

Un tableau contient les entités à classer. La méthode consiste à balayer le tableau en "inversant" deux entités successives a et b si elles ne répondent pas à la condition a < b.

Le tableau sera balayé jusqu'à ce qu'il n'y ait plus aucune inversion dans un balayage complet.

Classement de nombres

```
10 REM Classement de nombres
11 :
20 DIM T(100) '100 Donnees au maximum
100 REM Acquisition des donnees
101 :
110 CLS:PRINT "          CLASSEMENT DE NOMBRES"
120 PRINT:PRINT "Entrez les donnees a classer"
130 PRINT "La saisie se terminera par 9999"
140 :
150 I=1:PRINT
160 PRINT"Donnee"I;:INPUT T(I)
170 IF T(I)<>9999 THEN I=I+1:GOTO 160
180 :
190 REM Classement
200 :
210 CLS
220 PRINT "Voulez-vous un classement croissant"
230 INPUT "ou decroissant (C/D)";R$
240 IF R$<>"C" AND R$<>"D" THEN 190
250 :
260 IN=0 'Indicateur d'inversion
270 FOR J=1 TO I-1
280   C=T(J):D=T(J+1)
290   IF R$="C" AND C>D THEN T(J)=D:T(J+1)=C:IN=1
300   IF R$="D" AND C<D THEN T(J)=D:T(J+1)=C:IN=1
310 NEXT J
320 IF IN=1 THEN 260
330 :
400 REM Affichage des resultats
401 :
410 CLS:PRINT "           Donnees  classees":PRINT
420 FOR J=1 TO I
430   A=T(J)
440   IF A<>9999 THEN PRINT A
450 NEXT J
```

Classement de chaînes

```
10 REM Classement de chaines
11 :
20 DIM T$(100) '100 Donnees au maximum
100 REM Acquisition des donnees
101 :
110 CLS:PRINT "          CLASSEMENT DE CHAINES
"
120 PRINT:PRINT "Entrez les donnees a classer
"
130 PRINT "La saisie se terminera par 9999"
140 :
150 I=1:PRINT
160 PRINT "Donnee"I;:INPUT T$(I)
170 IF T$(I) <> "9999" THEN I=I+1:GOTO 160
180 :
190 REM Classement
200 :
210 CLS
220 PRINT "Voulez-vous un classement croissan
t"
230 INPUT "ou decroissant (C/D)";R$
240 IF R$ <> "C" AND R$ <> "D" THEN 190
250 :
260 IN=0 'Indicateur d'inversion
270 FOR J=1 TO I-1
280   C$=T$(J):D$=T$(J+1)
290   IF R$="C" AND C$>D$ THEN T$(J)=D$:T$(J+
1)=C$:IN=1
300   IF R$="D" AND C$<D$ THEN T$(J)=D$:T$(J+
1)=C$:IN=1
310 NEXT J
320 IF IN=1 THEN 260
330 :
400 REM Affichage des resultats
401 :
410 CLS:PRINT "          Donnees classees":PRI
NT
420 FOR J=1 TO I
430   A$=T$(J)
440   IF A$<>"9999" THEN PRINT A$
450 NEXT J
```

Analyse du programme

Lignes 10 à 130 : Présentation du programme.
Lignes 150 à 170 : Acquisition des données et classement.
Ligne 290 : Algorithme de classement par ordre croissant.
Ligne 300 : Algorithme de classement par ordre décroissant.
Lignes 400 à 450 : Affichage des résultats.

Remarque : les programmes de classement de chaînes et de nombres sont différents. En effet, si nous ne faisions qu'un programme de classement, il opérerait sur les chaînes de caractères, puisque les chaînes peuvent aussi bien contenir des données alphanumériques que des données numériques. Malheureusement, la chaîne "21" est inférieure à la chaîne "3" (d'un point de vue alphanumérique), ce qui est faux d'un point de vue numérique (en effet, on a 21 > 3 !). On est donc contraint de faire un programme de classement alphanumérique et un programme de classement numérique.

CHAÎNAGE DE DEUX PROGRAMMES AVEC PASSAGE DE PARAMÈTRES

Lorsqu'un programme occupe une place très importante en mémoire, il est parfois nécessaire de le scinder en plusieurs programmes plus courts.

Ces programmes sont alors chaînés pour réaliser l'ensemble des tâches initialement prévues.

Le problème suivant se pose : les résultats calculés dans l'un des programmes peuvent être nécessaires dans un autre programme de la chaîne. Comment préserver ces résultats ? (L'instruction LOAD efface en effet toutes les valeurs des variables BASIC.)

A priori, deux solutions paraissent envisageables :

- Stocker les variables à sauvegarder dans des emplacements mémoire non écrasés par l'instruction LOAD.
- Stocker les variables à sauvegarder dans un fichier séquentiel qui sera lu par le programme chaîné.

La première solution peut sembler séduisante. Elle n'est cependant pas retenue ici à cause de sa non-universalité, et de sa difficulté de mise en œuvre.

La deuxième solution est développée dans ce paragraphe. Elle suppose que l'utilisateur dispose d'une unité de disquettes.

Supposons que le programme N calcule des valeurs qui sont nécessaires au programme N+1, et que le programme N+1 soit chaîné au programme N.

Nous allons prendre le cas très général dans lequel le programme N calcule :

- des chaînes,
- des entiers,
- des réels,
- des tableaux,

nécessaires au programme N+1.

Les deux programmes qui suivent sont à incorporer respectivement dans les programmes N et N+1. Ils sont appelés par GOSUB 10000.

Le premier, situé dans le programme N, sauvegarde les chaînes, entiers, réels et tableaux nécessaires.

Le second, situé dans le programme N+1, récupère les chaînes, entiers, réels et tableaux sauvegardés par le premier et efface le fichier créé par le premier.

Exemple d'appel dans le programme N

```
10 A$="CHAINE 1":B$="CHAINE 2"
20 E1%=12:E2%=324
30 R1=-12.54:R2=6546.546
40 T$(1)="AZERTY":T$(2)="QZERTY"
50 T(1)=3:T(2)=-8.5:T(3)=45
60 GOSUB 10000
70 END
```

1er programme

```
10000 REM ***********************************
10001 REM *                                 *
10002 REM *  PASSAGE DE PARAMETRES PAR FICHIER  *
10003 REM *                                 *
10004 REM ***********************************
10005 :
10010 OPEN"O",#1,"PASSAGE"
10020 PRINT #1,A$,B$ 'Chaines
10030 PRINT #1,E1%,E2% 'Entiers
10040 PRINT #1,R1,R2 'Reels
10050 PRINT #1,T$(1),T$(2) 'T$
10060 PRINT #1,T(1),T(2),T(3) 'T
10070 CLOSE #1
10080 :
10090 RETURN
```

Analyse du programme

Ligne 10010 : Ouverture du fichier séquentiel “PASSAGE” en écriture.
Ligne 10020 : Ecriture de chaîne.
Ligne 10030 : Ecriture d'entiers.
Ligne 10040 : Ecriture de réels.
Ligne 10050 : Ecriture de tableau alphanumérique.
Ligne 10060 : Ecriture de tableau de réels.
Ligne 10070 : Fermeture du fichier “PASSAGE”.

Exemple d'appel dans le programme N+1

```
10 GOSUB 10000
20 PRINT A$,B$
30 PRINT E1%,E2%
40 PRINT R1,R2
50 PRINT T$(1),T$(2)
60 PRINT T(1),T(2),T(3)
70 :
80 END
```

2ᵉ programme

```
10000 REM **************************************
10001 REM *                                    *
10002 REM *LECTURE DE PARAMETRES DANS UN FICHIER*
10003 REM *                                    *
10004 REM **************************************
10005 :
10010 OPEN"I",#1,"PASSAGE"
10020 INPUT #1,A$,B$ 'Chaines
10030 INPUT #1,E1%,E2% 'Entiers
10040 INPUT #1,R1,R2 'Reels
10050 INPUT #1,T$(1),T$(2) 'T$
10060 INPUT #1,T(1),T(2),T(3) 'T
10070 CLOSE #1
10080 :
10090 RETURN
```

Analyse du programme

Ligne 10010 : Ouverture du fichier séquentiel “PASSAGE” en lecture.
Ligne 10020 : Lecture de chaîne.
Ligne 10030 : Lecture d'entiers.
Ligne 10040 : Lecture de réels.
Ligne 10050 : Lecture de tableau alphanumérique.
Ligne 10060 : Lecture de tableau de réels.
Ligne 10070 : Fermeture du fichier “PASSAGE”.

LES BONNES ADRESSES DES ROUTINES ASSEMBLEUR

Pour MO5

□ POKE &HA7C0,PEEK(&HA7C0) OR 1	Sélectionne la forme sur l'écran graphique.
□ POKE &HA7C0,PEEK(&HA7C0) AND 254	Sélectionne la couleur sur l'écran graphique.
□ PRINT CHR$(&H1B)+CHR$(&H79)	Produit un scroll lent.
□ PRINT CHR$(&H1B)+CHR$(&H78)	Produit un scroll normal.

Pour afficher le contenu d'un registre du micro-processeur, le charger dans le double registre D, puis faire JMP &D83E.

Le délai de répétition des touches se programme en &2076.

□ PEEK(&H201B)	Donne la ligne courante du curseur.
□ PEEK(&H201C)	Donne la colonne courante du curseur.

Pour TO7/70

□ PRINT CHR$(&H1B)+CHR$(&H6E)	Produit un scroll lent.
□ PRINT CHR$(&H1B)+CHR$(&H6A)	Produit un scroll normal.
□ PEEK(&H601B)	Donne la ligne courante du curseur.
□ PEEK(&H6020)	Donne la colonne courante du curseur.

Chapitre 2 — Améliorations des BASIC MO5 et TO7/70

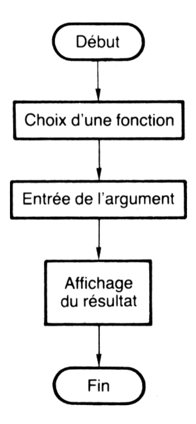
FONCTIONS SINUSOÏDALES ET HYPERBOLIQUES

MO5 et TO7/70 sont assez pauvres en fonctions sinusoïdales. En effet, ils possèdent seulement les fonctions **COS** (Cosinus), **SIN** (Sinus), et **TAN** (Tangente).

De plus, les fonctions hyperboliques usuelles sont totalement absentes.

Le programme qui suit lui ajoute les fonctions : Sécante, Cosécante, Cotangente, Sinus hyperbolique, Cosinus hyperbolique, Tangente hyperbolique, Sécante hyperbolique, Cosécante hyperbolique, Cotangente hyperbolique, Arc sécante hyperbolique, Arc cosécante hyperbolique, Arc sinus hyperbolique, Arc cosinus hyperbolique, Arc tangente hyperbolique, Arc cotangente hyperbolique.

Sa structure est très simple :



Programme

```
1 REM ******************************
2 REM *                            *
3 REM *  FONCTIONS TRIGONOMETRIQUES  *
4 REM *                            *
5 REM ******************************
6 :
20 CLS:PRINT"Choisissez votre fonction :"
30 PRINT:PRINT"1 )Secante"
40 PRINT"2 )Cosecante"
50 PRINT"3 )Cotangente"
90 PRINT"4 )Sinus Hyperbolique"
100 PRINT"5 )Cosinus Hyperbolique"
110 PRINT"6 )Tangente Hyperbolique"
120 PRINT"7 )Secante Hyperbolique"
130 PRINT"8 )Cosecante Hyperbolique"
140 PRINT"9 )Cotangente Hyperbolique"
150 PRINT"10)Arc Sinus Hyperbolique"
160 PRINT"11)Arc Cosinus Hyperbolique"
170 PRINT"12)Arc Tangente Hyperbolique"
180 PRINT"13)Arc Secante Hyperbolique"
190 PRINT"14)Arc Cosecante Hyperbolique"
200 PRINT"15)Arc Cotangente Hyperbolique"
210 PRINT:INPUT"Fonction choisie ";F
220 PRINT:INPUT"Argument ";A
230 ON F GOSUB 300,310,320,330,340,350,360,37
0,380,390,400,410,420,430,440
240 PRINT:PRINT"Resultat :";X
250 END
260 REM ----------------------------
300 REM Calcul Secante
301   X=1/COS(A)
302 RETURN
303 REM ----------------------------
310 REM Calcul Cosecante
311   X=1/SIN(A)
312 RETURN
313 REM ----------------------------
320 REM Calcul Cotangente
321   X=1/TAN(A)
322 RETURN
323 REM ----------------------------
330 REM Calcul Sinus Hyperbolique
331   X=(EXP(A)-EXP(-A))/2
332 RETURN
333 REM ----------------------------
340 REM Calcul Cosinus Hyperbolique
341   X=(EXP(A)+EXP(-A))/2
342 RETURN
```

```
343 REM -----------------------------
350 REM Calcul Tangente Hyperbolique
351   X=-EXP(A)/(EXP(A)+EXP(-A))*2+1
352 RETURN
353 REM -----------------------------
360 REM Calcul Secante Hyperbolique
361   X=2/(EXP(A)+EXP(-A))
362 RETURN
363 REM -----------------------------
370 REM Calcul Cosecante Hyperbolique
371   X=2/(EXP(A)-EXP(-A))
372 RETURN
373 REM -----------------------------
380 REM Calcul Cotangente Hyperbolique
381   X=EXP(-A)/(EXP(A)-EXP(-A))*2+1
382 RETURN
383 REM -----------------------------
390 REM Calcul Sinus Hyperbolique inverse
391   X=LOG(A+SQR(A*A+1))
392 RETURN
393 REM -----------------------------
400 REM Calcul Cosinus Hyperbolique Inverse
401   X=LOG(A+SQR(A*A-1))
402 RETURN
403 REM -----------------------------
410 REM Calcul Tangente Hyperbolique Inverse
411   X=LOG((1+A)/(1-A))/2
412 RETURN
413 REM -----------------------------
420 REM Calcul Secante Hyperbolique Inverse
421   X=LOG((SQR(-A*A+1)+1))/A
422 RETURN
423 REM -----------------------------
430 REM Calcul Cosecante Hyperbolique Inverse
431   X=LOG(SGN(A)*SQR(A*A+1)+1)/A
432 RETURN
433 REM -----------------------------
440 REM Calcul Cotangente Hyperbolique Invers
e
441   X=LOG((A+1)/(A-1))/2
442 RETURN
```

Analyse du programme

Lignes 20 à 220	: Menu.
Ligne 230	: Déroutement sur la procédure correspondant à la fonction choisie.
Ligne 240	: Affichage du résultat.
Lignes 300 à 442	: Routines de calcul des fonctions.

OPÉRATIONS EN DOUBLE PRÉCISION

Addition en double précision

Définition : Le BASIC MO5 et TO7/70 permet de faire des additions de nombres réels avec un nombre total de chiffres (avant et après la virgule) n'excédant pas 6.
Ce programme permet de faire des additions sur des nombres réels positifs avec 6 chiffres au maximum avant la virgule, et 6 chiffres au maximum après la virgule, d'où le nom d'addition en double précision.

Programme : Les deux nombres positifs à additionner sont placés dans les chaînes A$ et B$.
Le résultat en double précision est exprimé dans la chaîne C$.
Le programme est appelé par GOSUB 10000.

Exemple

```
10 A$="87654.56":B$="765642.47854"
20 GOSUB 10000
30 PRINT C$ 'Resultat de l'addition
40 END
```

Programme

```
10000 REM ********************************
10001 REM *                              *
10002 REM * ADDITIONS EN DOUBLE PRECISION*
10003 REM *                              *
10004 REM ********************************
10005 REM *                              *
10006 REM * Entree  : A$, B$ Nombres a   *
10007 REM *           additionner        *
10008 REM * Sortie  : C$=A$+B$           *
10009 REM *                              *
10010 REM ********************************
```

```
10011 :
10020 REM Extraction Partie Entiere, Partie D
ecimale
10030 :
10040 J=0:E1=0:D1=0
10050 FOR I=1 TO LEN(A$)
10060   IF MID$(A$,I,1)="." THEN J=I
10070 NEXT I
10075 IF J=0 THEN J=LEN(A$)+1
10080 FOR I=1 TO J-1
10090   X=ASC(MID$(A$,I,1))-48
10095   E1=E1+X*10^(J-I-1)
10100 NEXT I
10105 IF J-1=LEN(A$) THEN D1=0:GOTO 10140
10110 FOR I=J+1 TO LEN(A$)
10120   D1=D1+(ASC(MID$(A$,I,1))-48)*10^(LEN(
A$)-I)
10130 NEXT I
10140 J=0:E2=0:D2=0
10150 FOR I=1 TO LEN(B$)
10160   IF MID$(B$,I,1)="." THEN J=I
10170 NEXT I
10180 FOR I=1 TO J-1
10190   X=ASC(MID$(B$,I,1))-48
10195   E2=E2+X*10^(J-I-1)
10200 NEXT I
10205 IF J-1=LEN(B$) THEN D2=0:GOTO 10240
10210 FOR I=J+1 TO LEN(B$)
10220   D2=D2+(ASC(MID$(B$,I,1))-48)*10^(LEN(
B$)-I)
10230 NEXT I
10235 L1=LEN(STR$(INT(D1))):L2=LEN(STR$(INT(D
2)))
10236 IF L1>L2 THEN D2=D2*10^(L1-L2)
10237 IF L2>L1 THEN D1=D1*10^(L2-L1)
10238 D1=INT(D1):D2=INT(D2)
10240 :
10250 REM Addition
10260 :
10280 D$=STR$(INT(D1+D2))
10290 IF LEN(D$)=L1 OR LEN(D$)=L2 THEN R=0:GO
TO 10310
10300 R=1 'Mise a 1 de la retenue
10305 D$=RIGHT$(D$,LEN(D$)-1-R)
10310 :
10320 G$=STR$(INT(E1+E2+R))
10330 C$=G$+"."+D$
10340 :
10350 RETURN
```

Analyse du programme

Lignes 10020 à 10240 : Extraction des parties entières et décimales des nombres à additionner.
Lignes 10250 à 10350 : Addition.

Remarque : pour ne pas présenter un programme trop complexe, seule l'addition de nombres réels positifs a été envisagée. Le lecteur pourra, à moindres frais, transformer ce programme pour permettre l'addition de nombres positifs ou négatifs. Enfin, en partant du même principe (manipulation de chaînes), il sera intéressant d'écrire des programmes de soustraction, multiplication, division, extraction de racine carrée, etc.

CHANGEMENT DE BASE

Définition : Base = Nombre d'unités d'ordre n nécessaires pour former une unité d'ordre n+1. Ainsi, en base N, les nombres s'échelonnent de 0 à N−1, puis de 10 à 1(N−1), etc.

Conversion base B → base 10

Définition : Un nombre de base B est converti en le même nombre exprimé en base 10.

Programme : Fournir le nombre en base B rangé par unités dans le tableau A.
Fournir la base B dans laquelle est exprimé ce nombre.
Le résultat est le même nombre exprimé en base 10 dans N.
Ce programme est appelé par GOSUB 10000.

Exemple

```
10 DIM A(100)
20 A(1)=2:A(2)=3:A(3)=1:B=7
30 GOSUB 10000 'Conversion
40 PRINT N 'Nombre converti
50 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM *        CHANGEMENT DE BASE        *
10003 REM *        BASE B -> BASE 10         *
10004 REM *                                  *
10005 REM ************************************
10006 REM *                                  *
10007 REM * Entree : A=Nombre a convertir    *
10008 REM *          B=Base de conversion    *
10009 REM * Sortie : N=Nombre converti       *
10010 REM *                                  *
10011 REM ************************************
10012 :
10020 FOR I=1 TO 100
10030   IF A(I)<>0 THEN J=I
10040 NEXT I
10050 :
10060 FOR I=1 TO J
10070   N=N+A(I)*B^(J-I)
10080 NEXT I
10090 :
10100 RETURN
```

Analyse du programme

Lignes 10020 à 10040 : Calcul du nombre de chiffres du nombre exprimé en base B.

Lignes 10060 à 10080 : Conversion.

Conversion base 10 → base B

Définition : Un nombre en base 10 est converti en le même nombre exprimé en base B.

Programme : Fournir le nombre N en base 10.
Fournir la base désirée B.
Le résultat est le même nombre exprimé en base B dans le tableau A où chaque case représente une unité.
Ce programme est appelé par GOSUB 10000.

Exemple

```
10 DIM A(100)
20 N=122:B=16:GOSUB 10000
30 FOR I=J TO 1 STEP -1
40   PRINT A(I);
50 NEXT I
60 END
```

Programme

```
10000 REM *************************************
10001 REM *                                   *
10002 REM *        CHANGEMENT DE BASE         *
10003 REM *        BASE 10 -> BASE B          *
10004 REM *                                   *
10005 REM *************************************
10006 REM *                                   *
10007 REM * Entree : N=Nombre a convertir     *
10008 REM *          B=Base de conversion     *
10009 REM * Sortie : A=Nombre converti        *
10010 REM *          J=Nombre de chiffres     *
10011 REM *          significatifs dans A     *
10012 REM *                                   *
10013 REM *************************************
10014 :
10020 R=N
10030 I=I+1
10040 A(I)=R-INT(R/B)*B:R=INT(R/B)
10050 IF R>=B THEN 10030
10060 A(I+1)=R
10070 FOR I=1 TO 100
10080   IF A(I)<>0 THEN J=I
10090 NEXT I
10100 :
10110 RETURN
```

Analyse du programme

Lignes 10020 à 10060 : Conversion.
Lignes 10070 à 10090 : Calcul du nombre de chiffres du nombre exprimé en base B.

Conversion base N → base P

Définition : Un nombre en base N est converti en le même nombre exprimé en base P.

Programme : Fournir le nombre en base N.
Fournir les bases désirées N et P.
Le résultat est le même nombre exprimé en base P dans le tableau B où chaque case représente une unité.
Ce programme est appelé par GOSUB 10000.

Exemple

```
10 DIM A(100),B(100)
20 A(1)=3:A(2)=3:A(3)=1:N=10:P=10:GOSUB 10000 'Co
nversion Base 8 -> Base 16
30 FOR I=J TO 1 STEP -1
40   PRINT B(I);
50 NEXT I
60 END
```

Programme

```
10000 REM *************************************
10001 REM *                                   *
10002 REM *         CHANGEMENT DE BASE        *
10003 REM *         BASE N -> BASE P          *
10004 REM *                                   *
10005 REM *************************************
10006 REM *                                   *
10007 REM * Entree : A=Nombre a convertir     *
10008 REM *          N=Base de depart         *
10009 REM *          P=Base d'arrivee         *
10010 REM * Sortie : B=Nombre converti        *
10011 REM *          J=Nombre de chiffres     *
10012 REM *          significatifs dans A     *
10013 REM *                                   *
10014 REM *************************************
10015 :
10020 REM Conversion Base N -> Base 10
10021 :
10030 FOR I=1 TO 100
10040   IF A(I)<>0 THEN J=I
10050 NEXT I
10060 :
10070 FOR I=1 TO J
10080   NI=NI+A(I)*N^(J-I)
```

```
10090 NEXT I
10100 :
10110 REM Conversion Base 10 -> Base P
10111 :
10120 R=NI:I=0
10130 I=I+1
10140 B(I)=INT(R-INT(R/P)*P):R=INT(R/P)
10150 IF R>=P THEN 10130
10160 B(I+1)=R
10170 FOR I=1 TO 100
10180   IF B(I)<>0 THEN J=I
10190 NEXT I
10200 :
10210 RETURN
```

Analyse du programme

Lignes 10020 à 10090 : Conversion base N → base 10.
Lignes 10110 à 10190 : Conversion base 10 → base P.

OPÉRATIONS SUR LES MATRICES

Les habitués du calcul matriciel seront bien déçus en constatant que le BASIC MO5 ou TO7/70 ne comporte aucune instruction sur ce sujet. Que ces gens-là se rassurent : voici un ensemble d'utilitaires qui vont permettre de manipuler simplement les matrices. Ces utilitaires sont élémentaires. Ils autorisent les opérations de base (addition de matrices, inversion de matrices, saisie de matrices, etc.) nécessaires dans toute résolution informatique d'un problème matriciel.

Tous les utilitaires sont regroupés dans un seul programme. Un exemple d'appel de ces utilitaires est donné après leur description détaillée.

MATCON

Définition : La matrice X (A, B) est remplie de constantes.

Programme : Fournir la dimension A, B de la matrice et la constante C.
Sortie : la matrice X (A, B) contient la constante C.
Ce programme est appelé par GOSUB 10000.

Analyse du programme

Lignes 10120 à 10160 : mise à C de la matrice X.

MATIDN

Définition : La matrice carrée X (A, A) est rendue égale à l'identité : la diagonale de X est formée de 1. Les autres éléments sont à 0.

Programme : Fournir la dimension A de la matrice X.
Sortie : la matrice X est rendue égale à l'identité.
Ce programme est appelé par GOSUB 11000.

Analyse du programme

Ligne 11130 : Si la ligne I = la colonne J, alors IDN (I, J) = 1.
Ligne 11140 : Si la ligne I <> la colonne J, alors IDN (I, J) = 0.

MATINPUT

Définition : MATINPUT est utilisé pour attribuer des valeurs entrées au clavier aux éléments d'une matrice A * B.

Programme : Fournir la dimension A, B de la matrice.
La matrice X (A, B) est lue.
Ce programme est appelé par GOSUB 12000.

Analyse du programme

Lignes 12110 à 12150 : Boucle de lecture de la matrice.

MATPRINT

Définition : Affichage des valeurs contenues dans une matrice A * B.

Programme : Fournir la dimension de la matrice A, B et la matrice à afficher.
Ce programme est appelé par GOSUB 13000.

Analyse du programme

Lignes 13110 à 13160 : Affichage de la matrice.

MATREAD

Définition : Permet de lire le contenu de la matrice X en consultant les DATA contenues dans le programme.

Programme : Fournir la dimension A, B de la matrice et les DATA correspondantes.
Sortie : la matrice X contient les éléments lus en DATA.
Ce programme est appelé par GOSUB 14000.

Analyse du programme

Lignes 14110 à 14150 : Lecture de la matrice X stockée en DATA.

MATTRN

Définition : Donne la transposée d'une matrice de dimension A * B.
Par cette transformation, la 1re ligne de X devient la 1re colonne de Y, la 2e ligne de X devient la 2e colonne de Y, etc.

Programme : Fournir la dimension de la matrice A, B et la matrice à transposer X.
Sortie : Y = matrice transposée de X = ^{t}X.
Ce programme est appelé par GOSUB 15000.

Analyse du programme

Lignes 15120 à 15160 : Transposition de la matrice.

MAT+

Définition : Additionne un à un les éléments de deux matrices de même dimensions.

Programme : Fournir la dimension des matrices à additionner A, B et les matrices à additionner X et Y.
Sortie : Z = X + Y.
Ce programme est appelé par GOSUB 16000.

Analyse du programme

Lignes 16120 à 16160 : Calcul de la matrice Z = X + Y.

MAT–

Définition : Soustrait un à un les éléments de deux matrices de même dimension.

Programme : Fournir la dimension des deux matrices à soustraire A, B et les deux matrices à soustraire X et Y.
Sortie : Z = X – Y.
Ce programme est appelé par GOSUB 17000.

Analyse du programme

Lignes 17120 à 17160 : Calcul de la matrice Z = X – Y.

MAT*

Définition : Multiplie un à un les éléments de deux matrices de même dimension.

Programme : Fournir la dimension A, B des matrices à multiplier et les matrices à multiplier.
Sortie : Z = X * Y.
Ce programme est appelé par GOSUB 18000.

Analyse du programme

Lignes 18120 à 18190 : Calcul de la matrice Z = X * Y.

MATINV

Définition : Une matrice carrée A * B peut être inversée si son déterminant est non nul. Dans ce cas, le résultat de cette fonction est la matrice inverse.

Programme : Fournir la dimension A de la matrice carrée et la matrice X à inverser.
Sortie : déterminant de la matrice. S'il est non nul, la matrice inverse.
Ce programme est appelé par GOSUB 19000.

Analyse du programme

Lignes 19130 à 19460 : Calcul de la matrice inverse.
Ligne 19320 : Affichage du message "Pas d'inverse".

Le programme suivant fait une démonstration d'exécution de chacune des neuf routines précédentes.

Exemple

```
10 REM Operations sur les Matrices
20 :
30 DIM X(10,10),Y(10,10),Z(10,10)
40 :
1000 REM Test de la fonction MATCON
1010 :
1020 A=3:B=2:C=52:GOSUB 10000
1030 CLS:PRINT"Test MATCON. C=52":PRINT
1040 FOR I=1 TO A
1050   FOR J=1 TO B
1060     PRINT X(I,J);
1070   NEXT J
1080   PRINT
1090 NEXT I
1100 :
1110 REM Test de la fonction MATIDN
1120 :
1130 A=6:GOSUB 11000
1140 PRINT:PRINT"Test MATIDN.":PRINT
1150 FOR I=1 TO A
1160   FOR J=1 TO A
1170     PRINT X(I,J);
1180   NEXT J
1190   PRINT
1200 NEXT I
1210 :
1220 REM Test de la fonction MATINPUT
1230 :
1240 PRINT:PRINT"Test MATINPUT.":PRINT
1250 A=4:B=3:GOSUB 12000
1260 :
1270 REM Test de la fonction MATPRINT
1280 :
1290 PRINT:PRINT"Test MATPRINT.":PRINT
1300 X(1,1)=1:X(1,2)=2:X(1,3)=3
1310 X(2,1)=4:X(2,2)=5:X(2,3)=6
1320 A=2:B=3:GOSUB 13000
1330 :
1340 REM Test de la fonction MATREAD
1350 :
1360 DATA 1,2,3,4,5,6,7,8
1370 A=2:B=4:GOSUB 14000
1380 PRINT:PRINT"Test MATREAD.":PRINT
1390 FOR I=1 TO A
1400   FOR J=1 TO B
```

```
1410      PRINT X(I,J);
1420    NEXT J
1430    PRINT
1440 NEXT I
1450 :
1460 REM Test de la fonction MATTRN
1470 :
1480 X(1,1)=1:X(1,2)=2:X(1,3)=3
1490 X(2,1)=4:X(2,2)=5:X(2,3)=6
1500 A=2:B=3:GOSUB 15000
1510 PRINT:PRINT"Test MATTRN:":PRINT
1520 FOR I=1 TO B
1530   FOR J=1 TO A
1540      PRINT Y(I,J);
1550   NEXT J
1560   PRINT
1570 NEXT I
1580 :
1590 REM Test de la fonction MAT+
1600 :
1610 X(1,1)=1:X(1,2)=2:X(1,3)=3
1620 X(2,1)=4:X(2,2)=5:X(2,3)=6
1630 Y(1,1)=6:Y(1,2)=5:Y(1,3)=4
1640 Y(2,1)=3:Y(2,2)=2:Y(2,3)=1
1650 A=2:B=3:GOSUB 16000
1660 PRINT:PRINT"Test MAT+":PRINT
1670 FOR I=1 TO A
1680   FOR J=1 TO B
1690      PRINT Z(I,J);
1700   NEXT J
1710   PRINT
1720 NEXT I
1730 :
1740 REM Test de la fonction MAT-
1750 :
1760 X(1,1)=1:X(1,2)=2:X(1,3)=3
1770 X(2,1)=4:X(2,2)=5:X(2,3)=6
1780 Y(1,1)=6:Y(1,2)=5:Y(1,3)=4
1790 Y(2,1)=3:Y(2,2)=2:Y(2,3)=1
1800 A=2:B=3:GOSUB 17000
1810 PRINT:PRINT"Test MAT-":PRINT
1820 FOR I=1 TO A
1830   FOR J=1 TO B
1840      PRINT Z(I,J);
1850   NEXT J
1860   PRINT
1870 NEXT I
1880 :
1890 REM Test de la fonction MAT*
1900 :
1910 X(1,1)=1:X(1,2)=2:X(1,3)=3
```

```
1920 X(2,1)=4:X(2,2)=5:X(2,3)=6
1930 Y(1,1)=6:Y(1,2)=5
1940 Y(2,1)=4:Y(2,2)=3
1950 Y(3,1)=2:Y(3,2)=1
1960 A=2:B=3:GOSUB 18000
1970 PRINT:PRINT"Test MAT*":PRINT
1980 FOR I=1 TO A
1990   FOR J=1 TO A
2000     PRINT Z(I,J);
2010   NEXT J
2020   PRINT
2030 NEXT I
2040 :
2050 REM Test de la fonction MATINV
2060 :
2070 X(1,1)=3 :X(1,2)=-0.5:X(1,3)=-0.5
2080 X(2,1)=-1:X(2,2)=0   :X(2,3)=1
2090 X(3,1)=-1:X(3,2)=0.5 :X(3,3)=-0.5
2100 A=3:GOSUB 19000
2110 PRINT:PRINT"Test MATINV.":PRINT
2120 FOR I=1 TO A
2130   FOR J=1 TO A
2140     PRINT Y(I,J);
2150   NEXT J
2160   PRINT
2170 NEXT I
2180 :
2190 END
```

Programme regroupant les utilitaires

```
10000 REM ***************************************
10010 REM *                                     *
10020 REM *          FONCTION   MATCON          *
10030 REM *                                     *
10040 REM ***************************************
10050 REM *                                     *
10060 REM * Entree : A,B Dimension de la matrice*
10070 REM *          C   Constante              *
10080 REM * Sortie : X   Matrice forcee a C     *
10090 REM *                                     *
10100 REM ***************************************
10110 :
10120 FOR I=1 TO A
10130   FOR J=1 TO B
10140     X(I,J)=C
10150   NEXT J
10160 NEXT I
10170 RETURN
```

```
11000 REM ***************************************
11010 REM *                                     *
11020 REM *          FONCTION   MATIDN          *
11030 REM *                                     *
11040 REM ***************************************
11050 REM *                                     *
11060 REM * Entree : A Dimension de la matrice  *
11070 REM * Sortie : X Matrice Identite         *
11080 REM *                                     *
11090 REM ***************************************
11100 :
11110 FOR I=1 TO A
11120   FOR J=1 TO A
11130     IF I=J  THEN X(I,J)=1
11140     IF I<>J THEN X(I,J)=0
11150   NEXT J
11160 NEXT I
11170 RETURN
12000 REM ***************************************
12010 REM *                                     *
12020 REM *          FONCTION   MATINPUT        *
12030 REM *                                     *
12040 REM ***************************************
12050 REM *                                     *
12060 REM * Entree : A,B Dimension de la matrice*
12070 REM * Sortie : X   Matrice lue            *
12080 REM *                                     *
12090 REM ***************************************
12100 :
12110 FOR I=1 TO A
12120   FOR J=1 TO B
12130     PRINT"Ligne"I"Colonne"J;:INPUT X(I,J)
12140   NEXT J
12150 NEXT I
12160 RETURN
13000 REM ***************************************
13010 REM *                                     *
13020 REM *          FONCTION   MATPRINT        *
13030 REM *                                     *
13040 REM ***************************************
13050 REM *                                     *
13060 REM * Entree : A,B Dimension de la matrice*
13070 REM * Sortie : X   Matrice affichee       *
13080 REM *                                     *
13090 REM ***************************************
13100 :
13110 FOR I=1 TO A
13120   FOR J=1 TO B
13130     PRINT X(I,J);
13140   NEXT J
```

```
13150   PRINT
13160 NEXT I
13170 RETURN
14000 REM ***************************************
14010 REM *                                     *
14020 REM *           FONCTION   MATREAD        *
14030 REM *                                     *
14040 REM ***************************************
14050 REM *                                     *
14060 REM * Entree : A,B Dimension de la matrice*
14070 REM * Sortie : X   Matrice lue            *
14080 REM *                                     *
14090 REM ***************************************
14100 :
14110 FOR I=1 TO A
14120   FOR J=1 TO B
14130     READ X(I,J)
14140   NEXT J
14150 NEXT I
14160 RETURN
15000 REM ***************************************
15010 REM *                                     *
15020 REM *           FONCTION   MATTRN         *
15030 REM *                                     *
15040 REM ***************************************
15050 REM *                                     *
15060 REM * Entree : A,B Dimension de la matrice*
15070 REM *          Matrice a transposer       *
15080 REM * Sortie : Y   Matrice transposee     *
15090 REM *                                     *
15100 REM ***************************************
15110 :
15120 FOR I=1 TO A
15130   FOR J=1 TO B
15140     Y(J,I)=X(I,J)
15150   NEXT J
15160 NEXT I
15170 RETURN
16000 REM ***************************************
16010 REM *                                     *
16020 REM *           FONCTION   MAT+           *
16030 REM *                                     *
16040 REM ***************************************
16050 REM *                                     *
16060 REM * Entree : A,B Dimension des matrices *
16070 REM *          X,Y Matrices a additionner *
16080 REM * Sortie : Z = X + Y                  *
16090 REM *                                     *
16100 REM ***************************************
```

```
16110 :
16120 FOR I=1 TO A
16130   FOR J=1 TO B
16140     Z(I,J)=X(I,J)+Y(I,J)
16150   NEXT J
16160 NEXT I
16170 RETURN
17000 REM ***************************************
17010 REM *                                     *
17020 REM *          FONCTION   MAT-            *
17030 REM *                                     *
17040 REM ***************************************
17050 REM *                                     *
17060 REM * Entree : A,B Dimension des matrices *
17070 REM *          X,Y Matrices a soustraire  *
17080 REM * Sortie : Z = X - Y                  *
17090 REM *                                     *
17100 REM ***************************************
17110 :
17120 FOR I=1 TO A
17130   FOR J=1 TO B
17140     Z(I,J)=X(I,J)-Y(I,J)
17150   NEXT J
17160 NEXT I
17170 RETURN
18000 REM ***************************************
18010 REM *                                     *
18020 REM *          FONCTION   MAT*            *
18030 REM *                                     *
18040 REM ***************************************
18050 REM *                                     *
18060 REM * Entree : A,B Dimension des matrices *
18070 REM *          X,Y Matrices a multiplier  *
18080 REM * Sortie : Z = X * Y                  *
18090 REM *                                     *
18100 REM ***************************************
18110 :
18120 FOR I=1 TO A
18130   FOR J=1 TO B
18140     Z(I,J)=0
18150     FOR K=1 TO B
18160       Z(I,J)=Z(I,J)+X(I,K)*Y(K,J)
18170     NEXT K
18180   NEXT J
18190 NEXT I
18200 RETURN
18210 :
```

```
19000 REM ************************************
19010 REM *                                  *
19020 REM *          FONCTION   MATINV       *
19030 REM *                                  *
19040 REM ************************************
19050 REM *                                  *
19060 REM * Entree : A Dimension matrice     *
19070 REM *          X Matrice a inverser    *
19080 REM * Sortie : Y Matrice inversee      *
19090 REM *          R Determinant           *
19100 REM *                                  *
19110 REM ************************************
19120 :
19130 FOR I=1 TO A
19140   FOR J=1 TO A
19150     Y(I,J)=X(I,J)
19160   NEXT J
19170 NEXT I
19180 B=A-1
19190 FOR K=1 TO A
19200   P=Y(K,1)
19210   IF P<>0 THEN 19340
19220   FOR I=K+1 TO A
19240     IF Y(I,1)=0 THEN 19310
19250     FOR J=1 TO A
19260       R=Y(K,J)
19270       Y(K,J)=Y(I,J)
19280       Y(I,J)=R
19290     NEXT J
19300     GOTO 19200
19310   NEXT I
19320   PRINT"Pas d'inverse !!"
19330   GOTO 19470
19340   FOR J=1 TO B
19350     Y(K,J)=Y(K,J+1)/P
19360   NEXT J
19370   Y(K,A)=1/P
19380   FOR I=1 TO A
19390     IF I=K THEN 19450
19400     R=Y(I,1)
19410     FOR J=1 TO B
19420       Y(I,J)=Y(I,J+1)-R*Y(K,J)
19430     NEXT J
19440     Y(I,A)=-R*Y(K,A)
19450   NEXT I
19460 NEXT K
19470 RETURN
```

CHOIX DU DÉPART D'UNE SÉQUENCE PSEUDO-ALÉATOIRE

Lorsque les micro-ordinateurs MO5 et TO7/70 sont mis sous tension, l'appel à la fonction RND génère toujours le même nombre aléatoire. Ce programme permet de pallier ce défaut. Une boucle d'attente d'appui sur une touche du clavier incrémente en même temps le pointeur de nombre aléatoire. Le temps que l'opérateur mettra pour actionner la touche étant absolument aléatoire, le nombre aléatoire généré par la fonction RND le sera également !

Les lignes 140 à 160 sont à insérer dans tout programme se servant de la fonction RND.

Programme

```
100 REM Choix du depart d'une sequence pseudo
-aleatoire
110 :
120 CLS:PRINT "Appuyez sur une touche":PRINT
130 :
140 A$=INKEY$
150 J=RND
160 IF A$="" THEN 140
170 :
180 FOR I=1 TO 10
190   PRINT INT(RND*10);
200 NEXT I
210 :
220 END
```

Analyse du programme

Lignes 140 à 160 : Attente d'une action au clavier.
Lignes 180 à 200 : Affichage d'une séquence pseudo-aléatoire.

GESTION DE FICHIERS

Lorsque vous débranchez l'ordinateur, tout ce qui se trouvait en mémoire vive est perdu. Si vous désirez garder les données qui se trouvaient en mémoire vive, un moyen simple consiste à construire un fichier de données et à le stocker sur un support magnétique.

Ce programme gère des fichiers de données sur cassette ou disquette. Il réalise les opérations suivantes :

- création de structure,
- création de fiche possédant la structure définie,
- visualisation de l'ensemble des fiches,
- modification d'une fiche,
- suppression d'une fiche,
- lecture d'un fichier sur K7 ou disquette,
- écriture d'un fichier sur K7 ou disquette.

Programme

```
400 REM Initialisation
401 :
500 FIN=1 'Initialisation
501 DIM A$(20,100),LA$(20) '100 Fiches max, 2
0 articles/fiche max
502 :
503 REM ***********************************
**
1000 REM Menu principal
1010 :
1020 REM Titre
1030 :
1040   CLS:ATTRB 1,1:SCREEN 6,0,0:COLOR 1
1050   PRINT:PRINT"GESTION DE FICHIERS"
1060 :
1070 REM Menu
1080 :
1090   ATTRB 0,0:COLOR 6:LOCATE 0,6
1100   PRINT"Vous pouvez :":PRINT:PRINT
1110   PRINT"1 - Creer une structure,"
1120   PRINT"2 - Creer une fiche,"
1130   PRINT"3 - Visualiser l'ensemble des fi
ches,"
1140   PRINT"4 - Modifier une fiche,"
1150   PRINT"5 - Supprimer une fiche,"
1160   PRINT"6 - Lire sur disquette,"
1170   PRINT"7 - Sauvegarder sur disquette,"
1180   PRINT"8 - Sortir du programme."
1190   LOCATE 0,19
1200   PRINT"Votre choix (1..8) ?"
1210   B$=INPUT$(1):B=ASC(B$)-48
1220 :
1230   IF B<1 OR B>8 THEN 1000 'Reponse non v
alide
1240   ON B GOSUB 3000,3500,4000,4500,5000,55
00,6000,6500
```

```
1250 :
1260   IF FIN=1 THEN 1000 'Boucle principale
1270 :
1280 END
1290 REM ***********************************
***
3000 REM Creation de structure
3010 :
3020 REM Titre
3030 :
3040   CLS:COLOR 1
3050   PRINT"     CREATION DE STRUCTURE"
3060   COLOR 6:LOCATE 0,4
3070 :
3080 REM Creation
3090 :
3100   IF NA=0 THEN 3150
3110   PRINT"Structure deja existante."
3120   INPUT"On continue (O/N) ";R$
3130   IF R$="N" THEN 3240
3140 :
3150   INPUT"Nombre d'articles par fiche ";NA
:PRINT
3160 :
3170   FOR I=1 TO NA
3180     PRINT"Libelle article"I;:INPUT LA$(I
)
3190   NEXT I
3200 :
3210   PRINT"Structure creee."
3220   GOSUB 9000 'Message de fin de creation
3230 :
3240 RETURN
3250 REM ***********************************
***
3500 REM Creation de fiche
3510 :
3520 REM Titre
3530 :
3540   CLS:COLOR 1
3550   PRINT"    CREATION DE FICHE"
3560   COLOR 6:LOCATE 0,4
3570 :
3580 REM Creation
3590 :
3600   IF NA=0 THEN PRINT"Structure absente."
:GOTO 3690
3610 :
3620   INPUT "No d'identification ";NI
```

```
3625   IF A$(1,NI)<>"" THEN PRINT"Fiche deja
existante.":GOTO 3690
3630 :
3635   PRINT
3640   FOR I=1 TO NA
3650     PRINT LA$(I);:INPUT A$(I,NI)
3660   NEXT I
3670 :
3680   PRINT"Fiche en memoire."
3685   IF NI>NF THEN NF=NI
3690   GOSUB 9000 'Message de fin de creation
3700 :
3710 RETURN
3720 REM ***********************************
***
4000 REM Visualisation des fiches
4010 :
4020 REM Visualisation
4030 :
4040   FOR I=1 TO NF
4050     CLS:PRINT"Fiche"I:PRINT
4060     FOR J=1 TO NA
4070       PRINTLA$(J)"  : "A$(J,I)
4080     NEXT J
4090     GOSUB 9000 'Message intermediaire
4100   NEXT I
4110   :
4120   LOCATE 0,20:PRINT"Toutes les fiches on
t ete visualisees."
4140   GOSUB 9000 'Message de fin de visualis
ation
4150   :
4160 RETURN
4170 REM ***********************************
***
4500 REM Modification d'une fiche
4510 :
4520 REM Titre
4530 :
4540   CLS:COLOR 1
4550   PRINT"MODIFICATION D'UNE FICHE"
4560   COLOR 6:LOCATE 0,4
4570 :
4580 REM Modification
4590 :
4600   INPUT "No d'identification ";NI
4610 :
4620   IF A$(1,NI)<>"" THEN 4650
4630   PRINT"Cette fiche n'est pas en memoire
 ":GOTO 4780
```

```
4640    :
4650    REM Affichage de la fiche
4660    :
4670      FOR I=1 TO NA
4680        PRINT LA$(I)"  : "A$(I,NI)
4690      NEXT I
4700    :
4710    REM Saisie de la modification
4720    :
4730    LOCATE 0,5
4735    FOR I=1 TO NA
4740      PRINT LA$(I)"  ";:INPUT A$(I,NI)
4750    NEXT I
4760 :
4770    PRINT"Modification effectuee."
4780    GOSUB 9000 'mESSAGE DE FIN DE MODIFICA
TION
4790 :
4800 RETURN
4810 REM ***********************************
***
5000 REM Suppression d'une fiche
5010 :
5020 REM Titre
5030 :
5040    CLS:COLOR 1
5050    PRINT "SUPPRESSION D'UNE FICHE"
5060    COLOR 6:LOCATE 0,4
5070 :
5080 REM Suppression
5090 :
5100    INPUT "No d'identification ";NI
5110    :
5120    IF A$(1,NI)<>"" THEN 5150
5130    PRINT "Cette fiche n'est pas en memoir
e."
5140    :
5150    FOR I=1 TO NA
5160      PRINT LA$(I)"  : "A$(I,NI)
5170    NEXT I
5180    :
5190    PRINT:INPUT "Etes-vous sur (O/N) ";R$
5200    IF R$="N" THEN 5300
5210    :
5220    FOR I=1 TO NA
5230      A$(I,NI)=""
5240    NEXT I
5250    :
5260    PRINT "Fiche supprimee.":NF=NF-1
5261    :
```

```
5262    REM Suppression Physique
5263    :
5264    FOR I=NI TO NF
5265      FOR J=1 TO NA
5266        A$(J,I)=A$(J,I+1)
5267      NEXT J
5268    NEXT I
5269    FOR J=1 TO NA
5270      A$(J,I)=""
5271    NEXT J
5272    :
5280    GOSUB 9000 'Message de fin de suppress
ion
5290    :
5300 RETURN
5310 REM *********************************
***
5500 REM Lecture disquette ou cassette
5510 :
5520 REM Titre
5530 :
5540    CLS:COLOR 1
5550    PRINT"Lecture magnetique"
5560    COLOR 6:LOCATE 0,4
5570    :
5580 REM Lecture
5590 :
5600    PRINT"Cette commande detruira toute fi
che"
5610    PRINT"ou structure en memoire."
5620    PRINT:INPUT "On continue (O/N) ";R$
5630    IF R$="N" THEN 5840
5640    :
5650    INPUT"Nom du fichier ";NF$
5660    OPEN"I",#1,NF$
5670    :
5680    INPUT#1,NF,NA
5690    :
5700    FOR I=1 TO NA
5710      INPUT#1,LA$(I)
5720    NEXT I
5730    :
5740    FOR I=1 TO NF
5750      FOR J=1 TO NA
5760        INPUT#1,A$(J,I)
5770      NEXT J
5780    NEXT I
5790    :
5800    CLOSE #1
```

```
5810    :
5820    PRINT"Fichier en memoire."
5830    :
5840    GOSUB 9000 'Message de fin de lecture
5850    :
5860 RETURN
5870 REM ***********************************
***
6000 REM Ecriture disquette ou cassette
6010 :
6020 REM Titre
6030 :
6040    CLS:COLOR 1
6050    PRINT"   ECRITURE DISQUETTE OU CASSETT
E"
6060    COLOR 6:LOCATE 0,4
6070 :
6080 REM Ecriture
6090 :
6100    INPUT "Nom du fichier ";NF$
6110    :
6120    OPEN"O",#1,NF$
6130    :
6140    PRINT#1,NF,NA
6150    :
6160    FOR I=1 TO NA
6170      PRINT#1,LA$(I)
6180    NEXT I
6190    :
6200    FOR I=1 TO NF
6210      FOR J=1 TO NA
6220        PRINT#1,A$(J,I)
6230      NEXT J
6240    NEXT I
6250    :
6260    CLOSE #1
6270    PRINT"Fichier sauvegarde."
6280    :
6290    GOSUB 9000 'Message de fin de sauvegar
de
6300    :
6310 RETURN
6320 REM ***********************************
***
6500 REM Fin du Programme
6510 :
6520 REM Titre
6530 :
6540    CLS:COLOR 1
```

```
6550   PRINT"      SORTIE DU PROGRAMME"
6560   COLOR 6:LOCATE 0,4
6570 :
6580 REM Fin
6590 :
6600   PRINT"Cette commande detruira toute fi
che"
6610   PRINT"ou structure en memoire."
6620   PRINT:INPUT"On continue (O/N) ";R$
6630   IF R$="O" THEN FIN=-1 ELSE FIN=1
6640 :
6650 RETURN
6660 REM ***********************************
***
9000 REM Message de fin d'action
9010 :
9020   COLOR 3:PRINT
9030   PRINT"Pressez une touche pour continue
r"
9040   LOCATE 1,1,0 'Suppression du curseur
9050   COLOR 6
9060   C$=INPUT$(1) 'Attente de l'appui
9070 :
9080 RETURN
```

Analyse du programme

Lignes 400 à 503	: Initialisation.
Lignes 1000 à 1280	: Menu principal.
Lignes 3000 à 3240	: Création d'une structure.
Lignes 3500 à 3710	: Création d'une fiche.
Lignes 4000 à 4160	: Visualisation des fiches.
Lignes 4500 à 4800	: Modification d'une fiche.
Lignes 5000 à 5300	: Suppression d'une fiche.
Lignes 5500 à 5860	: Lecture K7 ou disquette.
Lignes 6000 à 6310	: Écriture K7 ou disquette.
Lignes 6500 à 6650	: Fin du programme.
Lignes 9000 à 9080	: Message de fin d'action.

HARD-COPY D'ÉCRAN BASSE RÉSOLUTION

Cet utilitaire permet de faire une copie d'écran en basse résolution sur une imprimante 80 colonnes. Il doit être inséré dans le programme qui nécessite la copie d'écran et appelé par GOSUB 10000.

Programme

```
10000 REM Hard-Copy d'ecran basse resolution
10010 :
10020 A$=INPUT$(1) 'Attente d'une action clavier
10030 :
10040 OPEN"O",#3,"LPRT:(80)"
10050 FOR I=0 TO 24
10060   FOR J=0 TO 39
10070     PRINT#3,CHR$(SCREEN(J,I));
10080   NEXT J
10090   PRINT#3
10100 NEXT I
10110 CLOSE#3
10120 :
10130 RETURN
```

Analyse du programme

Ligne 10020 : Attente d'une action sur le clavier pour commencer la copie d'écran.

Ligne 10070 : Copie d'un caractère.

CALENDRIER PERPÉTUEL

Cet utilitaire utilise la formule de Gauss pour établir la correspondance entre date chiffrée et jour de la semaine.

Après avoir entré la date (jour, mois et année) pour laquelle vous voulez connaître le jour, MO5/TO7 applique ces données à la formule de Gauss et affiche immédiatement le jour correspondant.

Exemple

```
10 JOUR=30:MOIS=5:AN$="1964":GOSUB 1000
20 CLS:PRINT"Le"JOUR"/"MOIS"/ "AN$" est un "L
J$
30 END
```

Programme

```
1000 REM ******************************
1001 REM *                            *
1002 REM *      CALENDRIER   PERPETUEL   *
1003 REM *                            *
1004 REM ******************************
1005 REM *                            *
1006 REM * Entree : JOUR=Jour          *
1007 REM *          MOIS=Mois          *
1008 REM *          AN$ =Annee         *
1009 REM * Sortie : LJ$ =Jour de la semaine*
1010 REM *                            *
1011 REM ******************************
1012 :
1020 REM Initialisation
1021 :
1030 FOR I=0 TO 6
1040   READ J$(I)
1050 NEXT I
1060 :
1070 DATA Dimanche,Lundi,Mardi,Mercredi,Jeudi
,Vendredi,Samedi
1080 :
1090 M=MOIS-2:IF MOIS<=2 THEN M=MOIS+10
1100 C=VAL(LEFT$(AN$,2)):A=VAL(RIGHT$(AN$,2))
1110 IF MOIS<=2 THEN A=A-1
1120 X=INT(2.6*M-0.199)+JOUR+A+INT(A/4)+INT(C
/4)-2*C
1130 Z=INT(X)-7*INT(X/7)
1140 LJ$=J$(Z)
1150 :
1160 RETURN
```

Analyse du programme

Lignes 1020 à 1070	: Initialisation.
Lignes 1090 à 1130	: Formule de Gauss.
Ligne 1140	: Résultat dans LJ$.

GESTION D'ERREURS

Ce petit utilitaire permet d'afficher en clair à l'écran le type d'une éventuelle erreur lors de l'exécution d'un programme. Il est à insérer en ligne 10000 et doit être initialisé par GOSUB 10000. Il est activé par :

ON ERROR GOTO 10100.

Toute erreur produira un arrêt du programme et un affichage de la cause de l'erreur.

Exemple

```
10 GOSUB 10000:ON ERROR GOTO 10100
20 INPUT#1,A$
30 END
```

Programme

```
10000 REM Traduction des erreurs MO5
10001 :
10010 DIM E$(61)
10020 FOR I=1 TO 61
10030   READ E$(I)
10040 NEXT I
10050 :
10060 DATA NEXT sans FOR correspondant,Erreur
 de syntaxe
10061 DATA RETURN sans GOSUB correspondant,RE
AD sans DATA correspondant
10062 DATA Argument incorrect dans une foncti
on,Nombre trop grand dans un calcul
10063 DATA Depassement de capacite en memoire
 centrale
10064 DATA Numero de ligne inexistant,Indice
en dehors des limites
10065 DATA Essai de re-dimensionnement d'un t
ableau,Division par zero
10066 DATA Instruction interdite en commande
directe
10067 DATA Nom de variable non correspondant
au type attendu
10068 DATA Manque de place pour les chaines,F
ormule de chaine trop complexe
10069 DATA Formule trop complexe,Suite de l'e
xecution impossible
10070 DATA Utilisation d'une fonction non def
inie,Instruction RESUME absente
10071 DATA RESUME sans ERROR correspondant,Er
reur non definie
10072 DATA Operande manquant,FOR sans NEXT co
rrespondant
10073 DATA ,,,,,,,,,,,,,,,,,,,,,,,,,
10074 DATA Numero de fichier incorrect,Mode d
'acces au fichier incorrect
```

```
10075 DATA Fichier deja ouvert,Erreur sur une
 Entree/Sortie
10076 DATA Essai de lecture d'un fichier apre
s fermeture
10077 DATA descripteur de fichier incorrect
10078 DATA commande directe dans un fichier e
n cours de chargement
10079 DATA Fichier non ouvert,Mauvaises donne
es dans un fichier
10080 DATA Peripherique occupe,Peripherique a
bsent ou indisponible
10081 DATA Programme protege
10082 :
10083 RETURN
10084 REM ************************************
****
10100 REM Gestion des Erreurs
10110 :
10120 PRINT E$(ERR);" Ligne";ERL
10130 RESUME NEXT
```

Analyse du programme

Lignes 10000 à 10083 : Définition du libellé des erreurs possibles.
Lignes 10100 à 10130 : Affichage du texte de l'erreur.

COPIE D'ÉCRAN MONOCHROME

Ce programme écrit en BASIC recopie en "semi-haute résolution" une partie de la mémoire d'écran, en faisant abstraction des couleurs. Pour activer ce programme, fournir ligne et colonne de début et de fin de recopie (la ligne peut varier entre 0 et 199, la colonne entre 0 et 39) et ligne et colonne de fin de recopie. Appeler l'utilitaire par GOSUB 10000.

Exemple

```
10 CLS
20 FOR I=1 TO 15
30   COLOR I:PRINT CHR$(64+I);
40 NEXT I
50 L1=0:C1=0:L2=7:C2=14:L3=100:C3=20:GOSUB 10
000
60 END
```

Programme

```
10000 REM ***********************************
****
10001 REM *
   *
10002 REM *    RECOPIE    D'ECRAN    COULEUR
   *
10003 REM *
   *
10004 REM ***********************************
****
10005 REM *
   *
10006 REM * Entree : L1,C1 Ligne et Colonne d
e  *
10007 REM *          debut de zone a recopier
   *
10008 REM *          L2,C2 Ligne et colonne d
e  *
10009 REM *          fin de zone a recopier
   *
10010 REM *          L3,C3 Ligne et colonne d
e  *
10011 REM *          debut de zone de recopie
   *
10012 REM *
   *
10013 REM ***********************************
****
10014 :
10020 POKE &HA7C0,PEEK(&HA7C0) OR 1:GOSUB 100
50 'Motif
10030 POKE &HA7C0,PEEK(&HA7C0) AND 254:GOSUB
10050 'Couleur
10040 RETURN
10045 :
10050 FOR I=L1 TO L2
10060   FOR J=C1 TO C2
10070     A=L3*40+C3+J-C1+(I-L1)*40
10080     B=J+I*40
10090     POKE A,PEEK(B)
10100   NEXT J
10110 NEXT I
10120 :
10130 RETURN
```

Analyse du programme

Lignes 10002 à 10100 : Recopie octet par octet.

Ligne 10060 : A = adresse de recopie.
B = adresse à recopier.

COPIE D'ÉCRAN COULEUR

Ce programme est identique au précédent, la recopie s'effectuant cette fois-ci en couleur. Mêmes paramètres en entrée que le programme précédent.

L'octet &HA7CO permet de sélectionner (MO5) la mémoire d'écran contenant le motif ou la couleur.

La copie d'écran s'effectue octet à octet :

- pour le motif,
- pour la couleur.

Exemple

```
10 CLS
20 FOR I=1 TO 15
30   COLOR I:PRINT CHR$(64+I);
40 NEXT I
50 L1=0:C1=0:L2=7:C2=14:L3=100:C3=20:GOSUB 10000
60 END
```

Programme

```
10000 REM ***************************************
10001 REM *                                     *
10002 REM *    RECOPIE    D'ECRAN    COULEUR    *
10003 REM *                                     *
10004 REM ***************************************
10005 REM *                                     *
10006 REM * Entree : L1,C1 Ligne et Colonne de  *
10007 REM *          debut de zone a recopier   *
10008 REM *          L2,C2 Ligne et colonne de  *
10009 REM *          fin de zone a recopier     *
10010 REM *          L3,C3 Ligne et colonne de  *
10011 REM *          debut de zone de recopie   *
10012 REM *                                     *
10013 REM ***************************************
10014 :
10020 POKE &HA7C0,PEEK(&HA7C0) OR 1:GOSUB 10050 'M
otif
10030 POKE &HA7C0,PEEK(&HA7C0) AND 254:GOSUB 10050
 'Couleur
10040 RETURN
10045 :
10050 FOR I=L1 TO L2
10060   FOR J=C1 TO C2
10070     A=L3*40+C3+J-C1+(I-L1)*40
10080     B=J+I*40
10090     POKE A,PEEK(B)
10100   NEXT J
10110 NEXT I
10120 :
10130 RETURN
```

Analyse du programme

Ligne 10020 : Sélection du motif.
Ligne 10030 : Sélection de la couleur.
Lignes 10050 à 10110 : Copie motif ou couleur selon sélection précédente.

INSERTION DE CHAÎNE

Ce programme permet d'insérer une chaîne de caractères dans une autre chaîne, à partir d'une position quelconque. Fournir la chaîne à insérer A$, la chaîne réceptrice B$, la position de début d'insertion P, C$ contiendra la chaîne B$ modifiée.

Exemple

```
10 B$="Il etait mieux":A$=" vraiment":P=8:GOS
UB 10000
20 PRINT C$
30 END
```

Programme

```
10000 REM **********************************
10001 REM *                                *
10002 REM *      FONCTION      INSERT      *
10003 REM *                                *
10004 REM **********************************
10005 REM *                                *
10006 REM * Entree : A$ Chaine a inserer   *
10007 REM *          B$ Chaine receptrice  *
10008 REM *          P  Position d'insertion*
10009 REM * Sortie : C$ Chaine B$ modifiee *
10010 REM *                                *
10011 REM **********************************
10012 :
10020 C$=LEFT$(B$,P) 'Debut
10030 C$=C$+A$ 'Insertion
10040 C$=C$+RIGHT$(B$,LEN(B$)-P) 'Fin
10050 :
10060 RETURN
```

Analyse du programme

Ligne 10020 : Extraction de B$ de la partie précédant l'insertion.
Ligne 10030 : Insertion.
Ligne 10040 : Rajout de la fin de B$ pour former la chaîne C$.

FONCTION RÉÉCRITURE

Ce programme permet de copier une chaîne A$ dans une chaîne B$ à partir d'une position donnée. Fournir la chaîne à recopier A$, la chaîne réceptrice B$, la position de début de recopie ; C$ contiendra la chaîne B$ modifiée en sortie du programme.

Exemple

```
10 B$="Babar l'elephant":A$="Bobor":P=0:GOSUB
 10000
20 PRINT C$
30 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM *        FONCTION    REECRITURE    *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : A$ Chaine a recopier    *
10007 REM *          B$ Chaine receptrice    *
10008 REM *          P  Position de recopie  *
10009 REM * Sortie : C$ Chaine B$ modifiee   *
10010 REM *                                  *
10011 REM ************************************
10012 :
10020 C$=LEFT$(B$,P) 'Debut
10030 C$=C$+A$ 'Recopie
10040 L=LEN(B$)-LEN(A$)-P
10050 IF L>0 THEN C$=C$+RIGHT$(B$,L) 'Fin
10060 :
10070 RETURN
```

Analyse du programme

Ligne 10020	: Extraction de B$ de la partie précédant l'écrasement.
Ligne 10030	: Écrasement de B$.
Ligne 10050	: Rajout de la fin de B$ pour former C$.

CENTRAGE DE TEXTE

Ce programme très simple s'avère très utile pour centrer une chaîne de longueur inférieure à 40 sur les 40 colonnes que comporte l'écran. Fournir en entrée le texte à centrer dans A$.

Exemple

```
10 A$="Essai de centrage":GOSUB 10000
20 A$="40 Colonnes":GOSUB 10000
30 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM *  CENTRAGE  DE  TEXTE  SUR  L'ECRAN  *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree  :  A$ Texte a centrer     *
10007 REM * Sortie  :  Affichage centre de A$    *
10008 REM *                                  *
10009 REM ************************************
10010 :
10020 L=LEN(A$)
10030 S=(40-L)/2 'Nombre d'espaces a inserer
10040 PRINT SPC(S)A$
10050 :
10060 RETURN
```

Analyse du programme

Ligne 10020	: Calcul de la longueur de A$.
Ligne 10030	: Calcul du décalage avant d'afficher la 1re lettre de A$.
Ligne 10040	: Affichage de A$ à la position courante curseur.

FONCTIONS MAX, MIN

Fonction MAX

Définition : Détermine laquelle de deux valeurs est la plus grande.

Exemple : Y = A max 5 donne à Y la valeur de A si A > 5, sinon donne à Y la valeur 5.

Programme : Fournir les deux nombres à comparer A et B.
Sortie : R = A MAX B
Ce programme est appelé par GOSUB 10000.

Exemple

```
10 A=5:B=8:GOSUB 10000
20 PRINT R 'R=8 car MAX(8,5)=8
30 END
```

Programme

```
10000 REM **************************************
10001 REM *                                    *
10002 REM *          FONCTION        MAX       *
10003 REM *                                    *
10004 REM **************************************
10005 REM *                                    *
10006 REM * Entree : A,B Donnees a comparer    *
10007 REM * Sortie : R = MAX(A,B)              *
10008 REM *                                    *
10009 REM **************************************
10010 IF A<B  THEN R=B
10020 IF A>=B THEN R=A
10030 :
10040 RETURN
```

Analyse du programme

Lignes 10010 à 10020 : Calcul de la fonction MAX.

Fonction MIN

Définition : Détermine laquelle de deux valeurs est la plus petite.

Exemple : Y = A MIN 2 donne à Y la valeur de A si A < 2,
donne à Y la valeur 2 si A > 2.

Programme : Fournir les deux nombres à comparer A et B.
Sortie : R = A MIN B.
Ce programme est appelé par GOSUB 10000.

Exemple

```
10 A=5:B=8:GOSUB 10000
20 PRINT R 'R=5 car MIN(8,5)=5
30 END
```

Programme

```
10000 REM *************************************
10001 REM *                                   *
10002 REM *          FONCTION       MIN       *
10003 REM *                                   *
10004 REM *************************************
10005 REM *                                   *
10006 REM * Entree : A,B Donnees a comparer   *
10007 REM * Sortie : R = MIN(A,B)             *
10008 REM *                                   *
10009 REM *************************************
10010 IF A>=B THEN R=B
10020 IF A<B  THEN R=A
10030 :
10040 RETURN
```

Analyse du programme

Lignes 10010 à 10020 : Calcul de la fonction MIN.

FONCTIONS DEEK, DOKE

Fonction DEEK

Définition : Donne le contenu de la mémoire sur deux octets (Double PEEK).

Programme : Fournir l'adresse du premier octet à connaître A.
Sortie : B = 1[er] octet, C = 2[e] octet.
Ce programme est appelé par GOSUB 10000.

Exemple

```
10 A=80 'Adresse du DEEK
20 GOSUB 10000
30 PRINT B,C
40 END
```

Programme

```
10000 REM ****************************************
10001 REM *                                      *
10002 REM *         FONCTION      DEEK           *
10003 REM *                                      *
10004 REM ****************************************
10005 REM *                                      *
10006 REM * Entree : Adresse du DEEK             *
10007 REM * Sortie : B=1er Octet                 *
10008 REM *          C=2eme Octet                *
10009 REM *                                      *
10010 REM ****************************************
10011 :
10020 B=PEEK(A)
10030 C=PEEK(A+1)
10040 :
10050 RETURN
```

Analyse du programme

Ligne 10020 : 1[er] octet dans B.
Ligne 10030 : 2[e] octet dans C.

Fonction DOKE

Définition : Place deux octets en mémoires consécutives (Double POKE).

Programme : Fournir l'adresse du 1[er] octet et les 2 octets à stocker.
Ce programme est appelé par GOSUB 10000.

Exemple

```
10 A=1000:B=&HAA:C=&HFF 'Adresse du DOKE et Data
20 GOSUB 10000
30 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM *         FONCTION     DOKE        *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : Adresse du DOKE         *
10007 REM *          B=1er Octet             *
10008 REM *          C=2eme Octet            *
10009 REM *                                  *
10010 REM ************************************
10011 :
10020 POKE A,B
10030 POKE A+1,C
10040 :
10050 RETURN
```

Analyse du programme

Ligne 10020 : Stockage du 1[er] octet.
Ligne 10030 : Stockage du 2[e] octet.

FONCTION D'AFFECTATION RÉPÉTITIVE : RPT$

Définition : Crée une chaîne contenant un certain nombre de caractères répétés plusieurs fois.

Programme : Fournir la chaîne à répéter : B$,
le nombre de répétitions : N.
Sortie : A$ = Chaîne B$ répétée N fois.

Exemple

```
10 B$="MINI":N=4:GOSUB 10000
20 PRINT A$ 'MINIMINIMINIMINI
30 :
40 B$="*":N=10:GOSUB 10000
50 PRINT A$ '**********
60 END
```

Programme

```
10000 REM ***************************************
10001 REM *                                     *
10002 REM * FONCTION DE REPETITION DE CHAINE    *
10003 REM *                                     *
10004 REM ***************************************
10005 REM *                                     *
10006 REM * Entree  : B$=Chaine source          *
10007 REM *           N =Nombre de repetitions  *
10008 REM * Sortie  : A$=Chaine repetee         *
10009 REM *                                     *
10010 REM ***************************************
10011 :
10020 A$="" 'RAZ Chaine resultat
10030 FOR I=1 TO N
10040   A$=A$+B$
10050 NEXT I
10060 RETURN
```

Analyse du programme

Lignes 10020 à 10050 : Fabrication de la chaîne A$ = RPT$(B$,N).

FONCTION DUMP : VISUALISATION DU CONTENU DE LA MÉMOIRE

Définition : Fournit le contenu hexadécimal d'une portion de mémoire.

Programme : Fournir le début et la fin de la mémoire dont on veut connaître le contenu.
Le programme liste le contenu de la mémoire, 8 octets/8.
Ce programme est lancé par RUN.

Programme

```
10000 REM ***********************************
10001 REM *                                 *
10002 REM *         DUMP        MEMOIRE     *
10003 REM *                                 *
10004 REM ***********************************
10005 :
10010 CLS:INPUT"<Debut>, <Fin>";D,F:LIGNE=D
10020 A=INT(LIGNE/256):GOSUB 10200:A$=B$
10025 A=LIGNE-A*256:GOSUB 10200:A$=A$+B$+"  "
10030 FOR I=0 TO 7
10040   A=PEEK(LIGNE+I)
10050   GOSUB 10200 'Dec -> Hex
10060   A$=A$+B$+" "
10070 NEXT I
10080 A$=A$+" "
10090 FOR I=0 TO 7
10100   A=PEEK(LIGNE+I)
10110   IF A<127 AND A>31 THEN A$=A$+CHR$(A):GOTO
10125
10120   A$=A$+"."
10125 NEXT I
10130 PRINT A$
10140 LIGNE=LIGNE+8
10150 IF LIGNE<=F THEN 10020
10160 END
10170 REM ******************************
10200 REM Sous-Programme de conversion
10210 REM Decimal -> Hexadecimal
10220 :
10230 REM Entree : A  en Decimal
10240 REM Sortie : B$ en Hexadecimal
10250 :
10260 B=INT(A/16):C=A-B*16
10270 IF B<=9 THEN B$=CHR$(B+48)
10280 IF B>9  THEN B$=CHR$(B+55)
10290 IF C<=9 THEN B$=B$+CHR$(C+48)
10300 IF C>9  THEN B$=B$+CHR$(C+55)
10310 RETURN
```

Analyse du programme

Ligne 10010 : Saisie du début et de la fin du DUMP.
Lignes 10020 à 10025 : Initialisations.
Lignes 10030 à 10150 : Calcul des données à afficher.
Lignes 10200 à 10310 : Conversion Décimal → Hexadécimal.

CONVERSION MINUSCULE→MAJUSCULE

Définition : Conversion d'une chaîne alphanumérique contenant des caractères minuscules en la même chaîne contenant des caractères majuscules.

Programme : Fournir A$ = Chaîne à convertir.
Sortie C$ = Chaîne convertie.

Exemple

```
10 A$="Basic Plus Copyright 1985"
20 GOSUB 10000
30 PRINT"Minuscule :"A$
40 PRINT"Majuscule :"C$
50 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM * CONVERSION minuscule -> MAJUSCULE *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : A$=Chaine a convertir   *
10007 REM * Sortie : C$=Chaine convertie     *
10008 REM *                                  *
10009 REM ************************************
10010 :
10020 FOR I=1 TO LEN(A$)
10030   B$=MID$(A$,I,1):B=ASC(B$)
10040   IF B<123 AND B>96 THEN C$=C$+CHR$(B-32)
10050   IF B>123 OR B<96 THEN C$=C$+B$
10060 NEXT I
10070 :
10080 RETURN
```

Analyse du programme

Ligne 10040 : Si le caractère est alphabétique, on le convertit.
Ligne 10050 : Sinon, on le laisse tel quel.

CONVERSION MAJUSCULE→MINUSCULE

Définition : Conversion d'une chaîne alphanumérique contenant des caractères majuscules en la même chaîne contenant des caractères minuscules.

Programme : Fournir A$ = Chaîne à convertir.
Sortie C$ = Chaîne convertie.

Exemple

```
10 A$="BASIC PLUS COPYRIGHT 1985"
20 GOSUB 10000
30 PRINT"Majuscule  :"A$
40 PRINT"Minuscule  :"C$
50 END
```

Programme

```
10000 REM ***************************************
10001 REM *                                     *
10002 REM * CONVERSION MAJUSCULE -> minuscule   *
10003 REM *                                     *
10004 REM ***************************************
10005 REM *                                     *
10006 REM * Entree  : A$=Chaine a convertir     *
10007 REM * Sortie  : C$=Chaine convertie       *
10008 REM *                                     *
10009 REM ***************************************
10010 :
10020 FOR I=1 TO LEN(A$)
10030   B$=MID$(A$,I,1):B=ASC(B$)
10040   IF B<91 AND B>64 THEN C$=C$+CHR$(B+32)
10050   IF B>90 OR B<65 THEN C$=C$+B$
10060 NEXT I
10070 :
10080 RETURN
```

Analyse du programme

Ligne 10040 : Si le caractère est alphabétique, on le convertit.
Ligne 10050 : Sinon, on le laisse tel quel.

FONCTION IN

Définition : Permet de savoir si un élément se trouve dans un ensemble.

Programme : Deux programmes sont fournis :
- un pour les chaînes de caractères ;
- un pour les nombres entiers ou réels.

Fournir l'ensemble A$ ou A,
le nombre d'éléments de l'ensemble,
l'élément à rechercher.

Sortie : R=−1 si l'élément se trouve dans l'ensemble, 0 sinon.

1er exemple

```
10 A$(1)="Toto":A$(2)="aime"
20 A$(3)="les" :A$(4)="gateaux"
30 N=4 '4 elements dans A$
40 B$="aime" 'Chaine recherchee
50 GOSUB 10000
60 PRINT R 'B$ IN A$
70 END
```

1er programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM *    FONCTION IN (VERSION CHAINE)  *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : A$=Ensemble de chaines  *
10007 REM *          N =Nb d'elements de A$  *
10008 REM *          B$=Chaine a comparer    *
10009 REM * Sortie : R = B$ IN A$            *
10010 REM *                                  *
10011 REM ************************************
10012 :
10020 R=0 'Preset Sortie
10030 FOR I=1 TO N
10040   IF B$=A$(I) THEN R=-1
10050 NEXT I
10060 :
10070 RETURN
```

Analyse du programme

Ligne 10020 : Valeur par défaut du résultat.
Lignes 10030 à 10050 : Fonction IN.

2e exemple

```
10 A(1)=1:A(2)=12:A(3)=-3
20 N=3 '3 elements dans A
30 B=13 'Element recherche
40 GOSUB 10000
50 PRINT R 'B IN A
60 END
```

2e programme

```
10000 REM ***************************************
10001 REM *                                     *
10002 REM *    FONCTION IN (VERSION NUMERIQUE)  *
10003 REM *                                     *
10004 REM ***************************************
10005 REM *                                     *
10006 REM * Entree : A=Ensemble de nombres      *
10007 REM *          N =Nb d'elements de A      *
10008 REM *          B =Nombre a comparer       *
10009 REM * Sortie : R = B IN A                 *
10010 REM *                                     *
10011 REM ***************************************
10012 :
10020 R=0 'Preset Sortie
10030 FOR I=1 TO N
10040   IF B=A(I) THEN R=-1
10050 NEXT I
10060 :
10070 RETURN
```

LE BASIC STRUCTURÉ

Certains types de BASIC, plus évolués que celui des MO5 et TO7/70, permettent d'employer des instructions évoluées de boucles du type :

REPEAT ... UNTIL ou encore DO ... WHILE.

L'instruction REPEAT ... UNTIL permet de répéter une instruction ou un groupe d'instructions jusqu'à ce qu'une condition soit réalisée.

Cette instruction peut être remplacée en BASIC Thomson par un branchement conditionnel.

A titre d'exemple, voici un programme utilisant REPEAT ... UNTIL écrit en BASIC structuré et sa traduction en BASIC Thomson.

BASIC structuré

```
10 REPEAT
20   I=I+1
30   PRINT I;
40 UNTIL I>5
50 END
```

BASIC MO5 et TO7/70

```
10 I=I+1
20 PRINT I;
30 IF I<=5 THEN 10
```

Analyse du programme

Ligne 30 : Simulation de l'ordre REPEAT.

L'instruction DO ... WHILE permet de répéter une instruction ou un groupe d'instructions tant qu'une condition est vérifiée.

Cette instruction peut être remplacée en BASIC Thomson par un branchement conditionnel.

A titre d'exemple, voici un programme écrit en BASIC structuré et utilisant DO ... WHILE et sa traduction en BASIC Thomson.

BASIC structuré

```
10 DO
20   I=I+1
30   PRINT I;
40 WHILE I<6
```

BASIC MO5 et TO7/70

```
10 I=I+1
20 PRINT I;
30 IF I<=5 THEN 10
```

Analyse du programme

Ligne 30 : Simulation de l'ordre DO.

JEUX DE CARACTÈRES MULTIPLES

Si, dans un même programme, vous désirez avoir plusieurs jeux de caractères, ce programme vous concerne. Les caractères sont définis en DATA. Un jeu peut comporter jusqu'à 128 caractères, chaque caractère étant défini par une suite de 8 octets. Le jeu sélectionné est mis dans la variable J et la procédure est appelée par GOSUB 10000.

Exemple

```
10 CLEAR,,128
20 J=1:GOSUB 10000
30 PRINT:PRINT"Jeu 1 : ";GR$(1)
40 J=2:GOSUB 10000
50 PRINT:PRINT"Jeu 2 : ";GR$(1)" "GR$(2)" "GR
$(3)
60 END
```

Programme

```
10000 REM **********************************
10001 REM *                                *
10002 REM *    JEUX DE CARACTERES MULTIPLES *
10003 REM *                                *
10004 REM **********************************
10005 REM *                                *
10006 REM * Entree : J=No du jeu choisi    *
10007 REM *                                *
10008 REM **********************************
10009 :
10010 RESTORE 13050
10020 I=0 'Initialisation
10030 I=I+1:IF I=J THEN 10070 'Fin de lecture
10040 READ A:FOR K=1 TO A*8:READ B:NEXT K
10050 GOTO 10030 'Boucle de lecture
10060 :
10070 READ A 'Nombre de DATA a redefinir
10080 FOR K=1 TO A
10090   FOR L=1 TO 8
10100     READ T(L)
10105     DEFGR$(K)=T(1),T(2),T(3),T(4),T(5),
T(6),T(7),T(8)
10110   NEXT L
10120 NEXT K
10130 :
12140 RETURN
13050 DATA 1
13060 DATA 1,2,4,8,&H10,&H20,&H40,&H80
13070 :
13150 DATA 3
13160 DATA &HFF,&H81,&H81,&H81,&H81,&H81,&H81
,&HFF
13170 DATA 1,2,4,8,&H10,&H20,&H40,&H80
13180 DATA &H80,&H40,&H20,&H10,8,4,2,1
```

Analyse du programme

Ligne 10	: Réservation de place pour 128 caractères au maximum.
Ligne 20	: Sélection du jeu 1.
Ligne 30	: Affichage des caractères du jeu 1.
Ligne 40	: Sélection du jeu 2.
Ligne 50	: Affichage des caractères du jeu 2.
Lignes 10010 à 10050	: Lecture des DATA non significatives.
Lignes 10070 à 10120	: Définition du jeu de caractères sélectionné.
Lignes 13050 à 13180	: DATA jeu 1 et jeu 2. Remarquez ligne 13050 DATA 1 signifie que le jeu 1 comporte 1 caractère, et ligne 13150 DATA 3 signifie que le jeu 2 comporte 3 caractères.

COULEUR D'UN CARACTÈRE DE L'ÉCRAN

Aucun ordre BASIC n'est prévu pour donner la couleur d'un caractère affiché à l'écran. Le programme suivant simule un tel ordre. Le caractère est repéré par ses coordonnées Ligne/Colonne (Ligne varie de 0 à 24, et Colonne de 0 à 39), et la procédure de recherche de couleur est appelée par GOSUB 10000. La couleur est donnée dans C à la sortie du programme. La couleur est 0 si aucun caractère n'est trouvé aux coordonnées LI,CO.

Exemple

```
10  CLS
20  FOR I=1 TO 15
30    COLOR I:PRINT CHR$(64+I);
40  NEXT I
50  LI=0:PRINT:PRINT:PRINT"Couleur des caract
eres :"
60  FOR I=1 TO 15
70    CO=I-1
80    GOSUB 10000:PRINT C;
90  NEXT I
100 END
```

Programme

```
10000 REM ***********************************
****
10001 REM *
   *
10002 REM *  COULEUR D'UN CARACTERE DE L'ECRA
N  *
10003 REM *
   *
10004 REM ***********************************
****
10005 REM *
   *
10006 REM * Entree : LI,CO Ligne et colonne
   *
10007 REM *            du test de couleur
   *
10008 REM : Sortie : C     Couleur du caracte
re *
10009 REM *
   *
```

```
10010 REM ***********************************
****
10011 :
10020 POKE &HA7C0,PEEK(&HA7C0)AND 254 'Scruta
tion memoire de couleur
10030 A=LI*40*8+CO
10040 C=PEEK(A)/16
10050 POKE &HA7C0,PEEK(&HA7C0)OR 1 'Retour a
la memoire d'ecan
10060 FOR B=A TO A+40*7 STEP 40
10070   IF PEEK(B)<> 0 THEN D=1
10080 NEXT B
10090 IF D=0 THEN C=0 'Pas de caractere
10100 :
10110 RETURN
```

Analyse du programme

Ligne 10020	: Sélection de la mémoire d'écran couleur.
Ligne 10030	: Calcul de l'adresse mémoire sélectionnée par LI/CO.
Ligne 10040	: Lecture couleur.
Lignes 10050 à 10090	: Test sur la validité de la couleur.

Chapitre 3 — Utilitaires d'écran

INPUT BORNÉ

Ce programme permet d'effectuer un INPUT numérique entier ou réel, compris entre deux bornes définies à l'avance. L'INPUT est refusé tant qu'il n'est pas compris entre les deux bornes.

L'appel au programme se fait par GOSUB 10000.

Au préalable,

- stocker dans L1 la valeur minimale permise,
- stocker dans L2 la valeur maximale permise,
- poser la question nécessitant l'input borné sous la forme PRINT "-" ; (lignes 20 à 30 dans l'exemple en n'oubliant pas le ";" en fin de message).

Exemple

```
10 L1=-10:L2=100 'Valeurs limites
20 CLS:PRINT"veuillez entrer un nombre"
30 PRINT"compris entre -10 et 100";
40 GOSUB 10000
50 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM *     REPONSE  NUMERIQUE  BORNEE   *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : L1 = Minimum            *
10007 REM *          L2 = Maximum            *
10008 REM * Sortie : N  = Nombre lu          *
10009 REM *                                  *
10010 REM ************************************
10011 :
10020 H=POS:V=CSRLIN 'Position du curseur
10030 :
10040 LOCATE H,V:PRINT SPC(15)
10050 LOCATE H,V:INPUT N
10060 IF N>L2 OR N<L1 THEN 10040
10070 :
10080 RETURN
```

Analyse du programme

Ligne 10020	: Lecture et mémorisation de la position du curseur.
Lignes 10040 à 10050	: Positionnement du curseur et INPUT.
Ligne 10060	: Test si le nombre entré est compris entre le minimum et le maximum.

SAISIE DE RÉPONSES PRÉDÉFINIES

Cet utilitaire permet de faciliter la saisie clavier de questions pour lesquelles le nombre de réponses est fini et restreint.

L'appui sur une touche quelconque permet de changer de réponse. La touche ENTREE valide la réponse choisie.

L'appel au programme se fait par GOSUB 10000.

Au préalable, mettre dans N le nombre de réponses possibles, dans A$ le libellé des réponses possibles. Poser la question sous la forme PRINT "-" (ligne 30 dans l'exemple).

Exemple

```
10 N=3 '3 valeurs Possibles
20 A$(1)="Oui":A$(2)="Non":A$(3)="Peut-etre"
'Valeurs Possibles
30 CLS:PRINT"Etes-vous d'accord ? ";
40 GOSUB 10000 'Gestion de la reponse
50 LOCATE 1,10,1:PRINT A$(S)" est votre repon
se"
60 END
```

Programme

```
10000 REM ************************************
****
10001 REM *
   *
10002 REM *   INPUT AVEC REPONSES PREDEFINIES
   *
10003 REM *
   *
10004 REM ************************************
****
10005 REM *
   *
10006 REM * Entree : N = Nb de valeurs possib
les*
10007 REM *            A$= Valeurs possibles
   *
10008 REM * Sortie : S = Valeur choisie
   *
10009 REM *
   *
10010 REM ************************************
****
10011 :
10020 REM Calcul de la longueur max de la rep
onse
10030 L=0 'Initialisation
10040 FOR I=1 TO N
10050   A=LEN(A$(I))
10060   IF A>L THEN L=A
10070 NEXT I
10080 :
10090 REM Initialisation
10100 :
10110 H=POS:V=CSRLIN 'Position du curseur
10120 LOCATE H,V,0:S=1:PRINT A$(S)
10130 :
10140 REM Saisie de la reponse
10150 :
10160 A$=INKEY$:IF A$="" THEN 10160
10170 IF ASC(A$)=13 THEN 10250 '<ENTREE>
10180 :
10190 REM Passage a la reponse suivante
10200 :
10210 S=S+1:IF S>N THEN S=1
10220 LOCATE H,V:PRINT A$(S);SPC(L-LEN(A$(S))
)
10230 GOTO 10160
10240 :
10250 REM <ENTREE>
10260 :
10270 RETURN
```

Analyse du programme

Lignes 10020 à 10070	: Calcul de la longueur maximale de la réponse.
Ligne 10110	: Lecture et mémorisation de la position du curseur.
Ligne 10160	: Lecture du clavier.
Ligne 10170	: Acceptation ou refus de l'entrée clavier.
Lignes 10190 à 10230	: Passage à la réponse suivante.

EFFACEMENT D'ÉCRAN

Ce programme permet d'effacer (remplir de blancs) tout ou partie de l'écran en mode texte. L'effacement est réalisé à partir d'une position X, Y du curseur et sur une longueur de N caractères. H et V contiennent respectivement l'abscisse et l'ordonnée de début de zone à effacer. N contient le nombre de caractères à effacer.

Exemple

```
10 H=10:V=10 'Position de depart d'effacement
20 N=400 'Nombre de caracteres a effacer
30 GOSUB 10000 'Effacement
40 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM * EFFACEMENT D'UNE PARTIE DE L'ECRAN *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : H=Position horizontale de *
10007 REM *          debut d'effacement       *
10008 REM *          V=Position verticake de  *
10009 REM *          debut d'effacement       *
10010 REM *          N=nombre de caractere(s) *
10011 REM *          a effacer                *
10012 REM *                                  *
10013 REM ************************************
10014 :
10150 LOCATE H,V
10160 IF N<255 THEN 10180
10170 PRINT SPC(255):N=N-255:GOTO 10160
10180 PRINT SPC(N)
10190 RETURN
```

Analyse du programme

Ligne 10150 : Positionnement du curseur en début de zone.
Ligne 10170 : Effacement modulo 255.
Ligne 10180 : Effacement résiduel.

ÉDITEUR DE MASQUES D'ÉCRAN

Pour faciliter la création d'"écrans de saisie", voici un utilitaire qui permet de fabriquer des "fichiers-écrans de saisie". Ces fichiers sont constitués d'un certain nombre de rubriques positionnées selon votre choix sur l'écran.

Programme

```
100 REM Editeur de masques d'ecran
110 :
120 CLS:PRINT SPC(7)"SAISIE DE MASQUES D'ECRA
N"
130 LOCATE 0,5
150 INPUT"Nombre de rubriques";NR
160 DIM A$(NR),C(NR),L(NR)
170 PRINT:INPUT"Titre du masque :";T$
180 :
190 FOR I=1 TO NR
200   CLS
210   COLOR 2,1:PRINT"Rubrique No";I;:COLOR 6
,0
220   LOCATE 0,5:INPUT"Libelle :";A$(I)
230   PRINT:INPUT"Colonne";C(I)
240   PRINT:INPUT"Ligne  ";L(I)
250 NEXT I
260 :
270 CLS:INPUT"Nom du fichier masque :";N$
280 OPEN "O",#1,N$
290 PRINT#1,NR 'Nombre de rubriques
300 PRINT#1,T$ 'Titre du masque
310 FOR I=1 TO NR
320   PRINT#1,A$(I) 'Rubrique
330   PRINT#1,L(I)  'Ligne d'affichage
340   PRINT#1,C(I)  'Colonne d'affichage
350 NEXT I
360 CLOSE #1
```

Analyse du programme

Lignes 100 à 130	: Présentation.
Ligne 150	: Nombre de rubriques dans le masque.
Ligne 170	: Titre du masque.
Lignes 190 à 250	: Saisie des positions et libellés des rubriques.
Lignes 270 à 360	: Écriture du fichier – masque.

EXPLOITATION DE FICHIERS MASQUES ÉCRAN

Cet utilitaire permet d'exploiter les masques d'écran créés par le programme précédent.

Le nom du fichier masque est inscrit dans la variable N$ et l'utilitaire est appelé par GOSUB 10000.

Les flèches "vers le bas" et "vers le haut" gèrent le déplacement parmi les rubriques. La rubrique sélectionnée apparaît en inverse vidéo. L'appui sur la touche ENTREE valide le choix de la rubrique.

Exemple

```
10 N$="MASQUE":GOSUB 10000
20 PRINT"Rubrique choisie :";P
30 END
```

Programme

```
10000 REM ***********************************
****
10001 REM *
   *
10002 REM *   AFFICHAGE ET GESTION DE MASQUES
   *
10003 REM *
   *
10004 REM ***********************************
****
10005 REM *
   *
10006 REM * Entree : N$ = Nom du fichier masq
ue *
10007 REM * Sortie : P  = Choix de la rubriqu
e  *
10008 REM *
   *
10009 REM ***********************************
****
```

```
10010 :
10011 SCREEN 6,0,0:LOCATE 1,1,0
10020 REM Lecture du masque
10030 :
10040 OPEN "I",#1,N$
10050 INPUT#1,NR 'Nombre de rubriques
10060 DIM A$(NR),C(NR),L(NR)
10070 INPUT#1,T$ 'Titre
10080 FOR I=1 TO NR
10090   INPUT#1,A$(I)
10100   INPUT#1,L(I)
10110   INPUT#1,C(I)
10120 NEXT I
10130 CLOSE #1
10140 :
10150 REM Affichage du masque
10160 :
10170 L=LEN(T$):L=(40-L)/2:CLS
10180 COLOR 6,1:LOCATE L,1:PRINT T$:COLOR 6,0
10190 FOR I=1 TO NR
10200   LOCATE C(I),L(I)
10210   PRINT A$(I)
10220 NEXT I
10230 :
10240 REM Gestion du masque
10250 :
10260 LOCATE C(1),L(1):COLOR 6,1:PRINTA$(1):C
OLOR 6,0:P=1
10270 A$=INKEY$:IF A$="" THEN 10270
10280 A=ASC(A$)
10290 IF A<>10 AND A<>11 AND A<>13 THEN 10270
10300 LOCATE C(P),L(P):PRINT A$(P)
10310 IF A=10 THEN 10390
10320 IF A=13 THEN 10450
10330 :
10340 REM Vers le haut
10350 :
10360 P=P-1:IF P=0 THEN P=NR
10370 GOTO 10430
10380 :
10390 REM Vers le bas
10400 :
10410 P=P+1:IF P>NR THEN P=1
10420 :
10430 LOCATE C(P),L(P):COLOR 6,1:PRINT A$(P):
COLOR 6,0:GOTO 10270
10440 :
10450 REM RETURN
10460 CLS
10470 RETURN
```

Analyse du programme

Lignes 10020 à 10130 : Lecture du masque N$.
Lignes 10150 à 10220 : Affichage du masque.
Lignes 10240 à 10470 : Gestion du masque.

- Lignes 10270 à 10320 : Gestion du clavier.
- Lignes 10340 à 10370 : Déplacement vers le haut.
- Lignes 10390 à 10410 : Déplacement vers le bas.
- Ligne 10430 : Inverse vidéo sur la rubrique sélectionnée.

SAISIE FORMATÉE

Peut-être vous êtes-vous déjà posé le problème suivant : est-il possible de saisir un nombre réel avec N chiffres avant la décimale et P chiffres après ? Cet utilitaire permet de le faire. Après avoir choisi le nombre de chiffres avant et après la virgule dans N et P, le masque de saisie est affiché sous la forme :



Le point décimal est géré automatiquement. La touche RETURN valide l'entrée du nombre. Le nombre stocké est entré dans la variable NL.

Exemple

```
10 N=2:P=4 '2 Chiffres avant la virg., 4 chif
fres apres
20 CLS:PRINT"Entrez un nombre reel :";
30 GOSUB 10000
40 END
```

Programme

```
10000 REM **********************************
10001 REM *                                *
10002 REM *       SAISIE NUMERIQUE FORMATEE *
10003 REM *                                *
10004 REM **********************************
10005 REM *                                *
10006 REM * Entree : N=Nombre de chiffres   *
10007 REM *          avant le point decimal *
10008 REM *          P=Nombre de chiffres   *
10009 REM *          apres le point decimal *
10010 REM * Sortie : NL=Nombre lu           *
10012 REM **********************************
10013 :
10020 REM Visualisation de l'entree
10030 :
10040 H=POS:V=CSRLIN 'Position du curseur
10050 LOCATE H,V+1
10060 IF N>0 THEN FOR I=1 TO N:PRINT"#";:NEXT
 I
10070 IF P<>0 THEN PRINT".";
```

```
10080 IF P>0 THEN FOR I=1 TO P:PRINT"#";:NEXT
 I
10090 LOCATE H,V
10100 :
10110 AV=1:I=0 'Initialisation
10120 A$=INKEY$:IF A$="" THEN 10120
10130 IF (A$>"9" OR A$<"0") AND A$<>"." AND A
SC(A$)<>13 AND A$<>"-" THEN PLAY"DO":GOTO 101
20
10140 I=I+1 'Nombre de caracteres entres
10150 IF A$="." THEN AV=0:I=0:PRINT".";:B$=B$
+A$:GOTO 10230
10160 IF ASC(A$)=13 THEN 10240
10170 IF AV=1 AND I<=N THEN B$=B$+A$:PRINT A$
;
10180 IF AV=1 AND I>=N AND P=0 THEN 10240
10190 IF AV=1 AND I>=N THEN B$=B$+".":PRINT".
";:AV=0:I=0:PLAY"DO":GOTO 10230
10200 IF AV=0 AND A$="-" THEN PLAY"DO":I=I-1:
GOTO 10230
10210 IF AV=0 AND I<=P THEN B$=B$+A$:PRINT A$
;
10220 IF AV=0 AND I>=P THEN PLAY"DO":GOTO 102
40
10230 GOTO 10120
10240 NL=VAL(B$):LOCATE H,V+1:PRINT SPC(N+P+1
)
10250 RETURN
```

Analyse du programme

Lignes 10020 à 10090 : Visualisation du masque de saisie.
Lignes 10110 à 10230 : Gestion du clavier.
Ligne 10240 : Stockage du nombre entré dans NL et effacement du masque de saisie.

AFFICHAGE PROGRAMMÉ

Cet utilitaire permet d'afficher du texte alphanumérique sur l'écran en mode texte. Le texte à afficher est inscrit dans le tableau A$. Les tableaux H et V donnent les positions horizontale et verticale du texte à afficher. Enfin, la variable N contient le nombre de libellés à afficher.

Exemple

```
10 REM Affichage Programme
20 :
30 N=3
40 H(1)=10:V(1)=3:H(2)=15:V(2)=15:H(3)=16:V(3)=18
50 A$(1)="Premier message"
60 A$(2)="Deuxieme message"
70 A$(3)="Troisieme message"
80 CLS:GOSUB 10000
90 END
```

Programme

```
10000 REM *************************************
10001 REM *                                   *
10002 REM *       AFFICHAGE   PROGRAMME       *
10003 REM *                                   *
10004 REM *************************************
10005 REM *                                   *
10006 REM * Entree : N=Nb de donnees a afficher *
10007 REM *          H=Positions Horizontales  *
10008 REM *          V=Positions Verticales    *
10009 REM *          A$=Chaine a afficher      *
10010 REM *                                   *
10011 REM *************************************
10012 :
10020 FOR I=1 TO N
10030   LOCATE H(I),V(I)
10040   PRINT A$(I)
10050 NEXT I
10060 RETURN
```

Analyse du programme

Ligne 10030 : Positionnement du curseur.
Ligne 10040 : Affichage du texte.

PERSONNALISEZ L'AFFICHAGE DE VOS MESSAGES A L'ÉCRAN

Pour donner une touche personnelle à l'affichage de vos programmes, voici un utilitaire qui allie son et affichage. L'affichage d'une lettre produit un "clic", et la durée entre l'affichage de deux lettres consécutives est choisi par le programmeur (fixe ou aléatoire).

Exemple

```
10 A$="Test du Programme MESSAGE SONORE"
20 TV=0:V=94:CLS:GOSUB 10000
30 END
```

Programme

```
10000 REM ************************************
****
10001 REM *
   *
10002 REM *      AFFICHAGE   PERSONNALISE
   *
10003 REM *
   *
10004 REM ************************************
****
10005 REM *
   *
10006 REM * Entree : TV=Vitesse d'affichage f
ixe*
10007 REM *          ou aleatoire (TV=0 ou TV
=1)*
10008 REM *          V=Vitesse d'affichage
   *
10009 REM *          entre 1 et 100
   *
10010 REM *
   *
10011 REM ************************************
****
10012 :
10013 PLAY"T1O4" 'Initialisation duree et hau
teur du son
10020 FOR I=1 TO LEN(A$)
10030   IF TV=1 THEN TEMPS=(100-V)*4 ELSE TEM
PS=(100-V)*RND*15
10040   B$=MID$(A$,I,1) 'Lettre a afficher
10050   FOR J=1 TO TEMPS:NEXT J 'Pause
10060   PRINT B$;
10070   PLAY"SO"
10080 NEXT I
10090 :
10100 RETURN
```

Analyse du programme

Ligne 10013	: Initialisation des durée et hauteur du clic sonore.
Ligne 10030	: Calcul du temps entre deux affichages.
Ligne 10040	: Extraction de la lettre à afficher.
Ligne 10050	: PAUSE.
Ligne 10060	: Affichage.
Ligne 10070	: Clic sonore.

SAISIE OPTIQUE

Pour clore ce chapitre sur les utilitaires écran, voici un programme qui permet de choisir dans un menu une des options proposées avec le crayon optique. Le nombre d'options doit être inférieur à 11. Le titre ou la question posé est mis dans A$, le nombre d'options possibles dans N et les différentes options dans T$. Le programme est appelé par GOSUB 10000.

Exemple

```
10 A$="ESSAI DE SAISIE OPTIQUE"
20 N=9:DIM T$(9)
30 T$(1)="Libelle 1"
40 T$(2)="Libelle 2"
50 T$(3)="Libelle 3"
60 T$(4)="Libelle 4"
70 T$(5)="Libelle 5"
80 T$(6)="Libelle 6"
90 T$(7)="Libelle 7"
100 T$(8)="Libelle 8"
110 T$(9)="Libelle 9"
120 GOSUB 10000
130 LOCATE 0,22:PRINT "Option choisie :";O
140 END
```

Programme

```
10000 REM ************************************
****
10001 REM *
   *
10002 REM *          SAISIE       OPTIQUE
   *
10003 REM *
   *
10004 REM ************************************
****
10005 REM *
   *
10006 REM * Entree : N = Nb d'options possibl
es *
10007 REM *          A$= Tableau des options
   *
10008 REM *          T$= Titre ou question
   *
10009 REM * Sortie : O = Option choisie
   *
10010 REM *
   *
10011 REM ************************************
****
10020 CLS:SCREEN 6,0,0 'Ecran noir Encre bleu
e
10030 A=INT(21/N)
10040 C=LEN(A$):C=(40-C)/2
10050 :
10060 LOCATE C,0,0:PRINT A$ 'Titre
10070 :
```

```
10080 FOR I=0 TO N-1
10090   BOX(0,I*A*8+32)-(20,I*A*8+42)
10100   BOXF(0,I*A*8+32)-(20,I*A*8+42)
10110   LOCATE 5,I*A+4:PRINT T$(I+1) 'Option
10120 NEXT I
10130 :
10140 INPUTPEN X,Y
10145 BEEP
10150 IF X=-1 OR Y=-1 THEN 10140 'Mauvaise en
tree
10160 :
10165 O=1 'Initialisation
10170 FOR I=1 TO N
10180    IF Y>I*A*8+32 AND Y< I*A*8+42 THEN O=
I+1
10190 NEXT I
10200 :
10210 RETURN
```

Analyse du programme

Lignes 10030 à 10040 : Calcul préliminaire.
Ligne 10060 : Affichage du titre.
Lignes 10080 à 10120 : Affichage des options possibles.
Lignes 10140 à 10190 : Gestion du crayon lumineux.

FONCTION DELAI

L'instruction PAUSE étant absente du BASIC MO5 ou TO7/70, voici un moyen de la simuler. L'instruction pause permet d'arrêter l'exécution du programme pendant un certain temps. Le temps d'arrêt en milliseconde(s) est placé dans la variable T et l'utilitaire est appelé par GOSUB 10000.

Exemple

```
10 T=100 '10 Secondes de Pause
20 GOSUB 10000:PLAY"DO"
30 END
```

Programme

```
10000 REM Pause
10001 :
10010 REM Entree T en 1/10 sec
10020 :
10030 T=58*T
10040 FOR W=1 TO T:NEXT W
10050 :
10060 RETURN
```

Analyse du programme

Lignes 10030 à 10040 : Boucle pendant Tms.

Chapitre 4 — Le générateur sonore

QU'EST-CE QU'UN SON ?

Pour répondre à cette question, faisons un parallèle simple : un caillou jeté dans une mare produit une onde qui a pour centre le point d'impact du caillou et qui s'en éloigne à vitesse constante. De même, un corps sonore qui a subi un choc émet un mouvement de vibration ou d'ondulation.

L'air qui entoure ce corps participe au mouvement, et forme autour de lui des ondes qui s'éloignent du corps à vitesse constante et qui parviennent à l'oreille.

Les sons perceptibles ont une fréquence de vibration comprise entre 16 et 15000 périodes/Sec (ou Hertz). On parlera d'infra-son pour une fréquence inférieure à 16 Hz, hertz et d'ultra-son pour une fréquence supérieure à 15000 Hz.

LA GÉNÉRATION DE SON SUR MO5 ET TO7/70

Le générateur de son des MO5 et TO7/70 se réduisent à leur plus simple expression puisqu'ils utilisent un vulgaire "Buzzer". En d'autres termes, le synthétiseur sonore (disons plutôt générateur sonore) est constitué d'un

haut-parleur sur lequel on envoie une fréquence variable. Le générateur sonore possède une voix. Un son quelconque est activé par une seule commande assez performante : PLAY.

Cette commande permet :

- de jouer une note DO DO# RE RE# MI FA FA# SOL SOL# LA LA# SI ;
- de changer d'octave O1 à O5 ;
- de changer la longueur d'émission d'une note ;
- de changer le nombre de notes émises par unité de temps T1 à T255 ;
- de changer l'attaque de la note A0 à A255.

Les programmes qui suivent mettent en œuvre ces diverses possibilités.

GAMME CHROMATIQUE

Ce programme montre les possibilités d'étendue en fréquence du générateur sonore. Il joue la gamme chromatique de l'octave 1 à l'octave 5.

Programme

```
10   REM Gamme chromatique
20
30   CLS:PRINT"Gamme chromatique :":PRINT
40   :
45   PLAY"L60" 'Initialisation
46   :
50   FOR I=1 TO 5 'Nombre d'octaves
60     O$="O"+RIGHT$(STR$(I),1):PLAY O$ 'Octav
e
70     COLOR 3:PRINT:PRINT"Octave ";I:PRINT:CO
LOR 6
80     RESTORE
90     FOR J=1 TO 12
100      READ A$:PRINT A$" ";:PLAY A$
110    NEXT J
120  NEXT I
130  :
140  END
150  :
160  DATA DO,DO#,RE,RE#,MI,FA,FA#,SO,SO#,LA,LA
#,SI
```

Analyse du programme

Ligne 30 : Titre.
Ligne 45 : Choix de la longueur d'émission d'une note.
Lignes 50 à 120 : Activation de la gamme sur 5 octaves.
  Ligne 60 : Activation d'une octave.
  Ligne 70 : Affichage de la note qui va être jouée.
  Lignes 90 à 110 : Activation de la gamme chromatique sur une octave.
Ligne 160 : DATA de la gamme chromatique.

MO5-PIANO

Si vous jouez du piano, ce programme peut vous sembler un peu désuet. Si vous n'en jouez pas, c'est l'occasion de vous y mettre...

Les touches du clavier reprennent la disposition des tons et 1/2 tons du clavier d'un piano. L'octave est sélectionnée grâce aux touches 1 à 5.

Programme

```
10 REM MO5-Piano
20 :
30 GOSUB 1000 'Initialisation
40 GOSUB 2000 'Piano
50 :
60 END
70 REM ************************************
1000 REM Initialisation des touches
1010 :
1020 DIM G$(26)
1030 FOR I=1 TO 26
1040   READ G$(I)
1050 NEXT I
1060 :
1070 DATA ,,,MI,RE#,FA,SO,LA,,SI,,,,,,,DO,,RE
,FA#,,,,,SO#,DO#
1080 :
1090 RETURN
1100 REM *********************************
2000 REM MO5-Piano
2010 :
2020 CLS:PRINT"MO5-Piano..."
2030 A$=INKEY$
2040 IF A$="" THEN 2030 'Boucle d'attente
2050 :
2060 A=ASC(A$)
2070 IF A=32 THEN 2160
2080 IF A>56 THEN 2100
2090 IF A<54 AND A>48 THEN O$="O"+RIGHT$(STR$
(A-48),1):PLAY O$ 'Octave
2100 IF A>90 OR A<65 THEN 2030 'Touche inconn
ue
2110 :
```

```
2120 IF G$(A-64)<>"" THEN PLAY G$(A-64)
2130 :
2140 GOTO 2030 'Boucle principale
2150 :
2160 RETURN
```

Analyse du programme

Lignes 10 à 60	: Programme principal.
Lignes 1000 à 1090	: Affectation d'une note ou d'un silence à chaque touche du clavier.
Lignes 2000 à 2160	: Activation des notes sélectionnées.
Lignes 2030 à 2040	: Lecture clavier.
Lignes 2070 à 2120	: Action en fonction de la touche pressée.

PROGRAMMATION DE MORCEAUX DE MUSIQUE

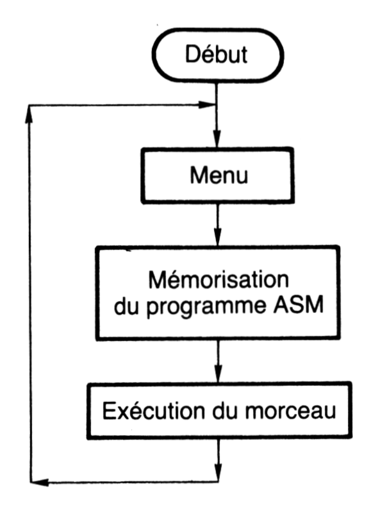
Le programme suivant met en évidence la facilité avec laquelle il est possible de transcrire un morceau dans l'hypothèse où l'on possède sa partition.

Cinq morceaux connus ont été choisis à cet effet : **La truite de Schubert, Pop Corn** et **Histoires sans parole, Classique** et **Midnight Express.**

Pour chacun d'eux, la partition a été traduite en notes et durées.

Ces notes et durées ont été mises sous la forme de DATA dans le programme.

La structure du programme est la suivante :



Programme

```
10 REM Themes Celebres
20 :
30 GOSUB 1000 'Menu
40 IF C$="E" THEN 80 'Fin
50 GOSUB 2000 'Activation morceau
60 GOTO  30   'Boucle principale
70 :
80 END
90 REM ********************************
1000 REM Menu
1010 :
1020 CLS:LOCATE 0,8:SCREEN 6,0,0:COLOR 1
1030 PRINT"Je peux jouer :":PRINT
1040 COLOR 6:PRINT"   ->1) Pop Corn,"
1050 PRINT"   ->2) Histoires sans paroles,"
1060 PRINT"   ->3) Classique,"
1070 PRINT"   ->4) Midnight express,"
1080 PRINT"   ->5) La truite de Shubert."
1090 :
1100 LOCATE 0,20:COLOR 3
1110 INPUT"Votre choix (1,2,3,4,5, ou E)nd) "
;C$
1120 :
1130 NP=0:A=VAL(C$)
1140 IF A=2 THEN NP=75
1150 IF A=3 THEN NP=152
1160 IF A=4 THEN NP=225
1170 IF A=5 THEN NP=260
1180 :
1190 RETURN
1200 REM ********************************
2000 REM Activation du morceau
2010 :
2020 REM Lecture des donnees non significativ
es
2030 :
2040 IF NP=0 THEN 2090
2050 FOR I=1 TO NP+1
2060   READ A$
2070 NEXT I
2080 :
2090 REM Activation du morceau
2100 :
2110 READ A$
2120 FOR I=1 TO VAL(A$)
2130   READ A$
2140   PLAY A$
2150 NEXT I
2160 :
2170 RESTORE 'RAJ pointeur de DATA
2180 :
2200 RETURN
2210 REM ********************************
3000 REM Pop Corn
3010 :
```

```
3020 DATA 75,O4L24MI,RE,MI,O3SI,SO,SI,MIO4P,M
I,RE,MI,O3SI,SO,SI,MIO4P,MI,FA#
3030 DATA SO,FA#,SO,MI,FA#,MI,FA#,RE,MI,RE,MI
,DO,MIP,L48SO,MI,O3SIO4
3040 DATA L24PSO,LA,L48SI
3050 DATA LA,SOP,SO,MI,O3SIO4L24P,SO,LA,L48SI
,LA,SO,L24P,MI,RE,MI,O3SI,SO
3060 DATA SI,MIO4P,MI
3070 DATA RE,MI,O3SI,SO,SI,MIO4P,MI,FA#,SO,FA
#,SO,MI,FA#,MI,FA#,RE,MI,RE,MI
3080 DATA DO,MI
3090 REM *********************************
3100 REM Histoires sans Parole
3110 :
3120 DATA 76,O5L12MI,DO,O4SO,MI,DO,MI,SO,MI,S
O,L24LAL12
3130 DATA MI,SOP,O5MI,DO,O4SO,MI,DO,MI,SO,MI
3140 DATA SO,O5DOO4,SI,O5RE,L24DOL12P,MI,L24S
OL12,MI,DOP,MI
3145 DATA DO,O4L24SOL12,LAP,O5MI,L24SOL12,MI,
DOP,MI,DO,O4SO
3150 DATA O5DO,DOO4,SI,O5DOP
3160 DATA MI,DO,O4SO,MI,DO,MI,SO,MI,SO,L24LAL
12
3170 DATA MI,SOP,O5MI,DO,O4SO,MI,DO,MI,SO,MI
3180 DATA SO,O5DOO4,SI,O5RE,DOO4P,L6SOP,SO,SO
,L12SO#,SOP,SO,O5DO
3190 REM *********************************
3200 REM Classique
3210 :
3220 DATA 72,O4L24RE,L12SO,FA#,L24SO,RE,LA,RE
,SIb,SO,O5DOO4,SO,O5REO4,SO
3230 DATA O5MIbO4,SO,L12FA#,SO,LA,FA#,L24REL1
2P,O5RE,DO,O4SI
3240 DATA O5DO,RE,O4SI,SO,LA,SI,SO,L24FA,L12M
Ib,RE,MIbP
3250 DATA O5DO,O4SIb,LA,SIb,O5DOO4,LAFA,SO,LA
,FA,L24MIbL12,RE,DO,REP
3260 DATA SIb,LA,SO,LA,SIb,SO,MIb,FA,SO,MIb,L
24REL12,DO,O3SIO4,DOP
3270 DATA L24MIb,RE,SO,L12FA#,SO,LA,FA#,L24FA
#,L12SO,FA#,L24SO
3280 REM *********************************
3300 REM Midnight Express
3310 :
3320 DATA 34,O5L24RE,DO,O4SO,MIb,RE,L36DOL24P
3330 DATA O5RE,DOO4,SO,MIb,RE,DO,RE,MIb
3340 DATA L48FA,L24SO,L96REL24,O5DOO4,SIb
3350 DATA LAb,MIb,FA,SO,LAb,SIb,L48O5DOO4
3360 DATA L24PLAb,SIb,O5DO,RE,MIb,FA,SO,L96DO
3370 REM *********************************
3400 REM La truite de Shubert
3410 :
3420 DATA 32,O4L24DO,FA,FA,LA,LA,L48FA,DO,L12
,SO,FA,MI,RE,L48DO
3430 DATA L12SO,FA,MI,RE,L48DOL24,DO,FA,FA,LA
,LA,L48FAL24,DO
3440 DATA FA,MI,L12RE,MIL24,FA,O3SIO4,DO
```

Analyse du programme

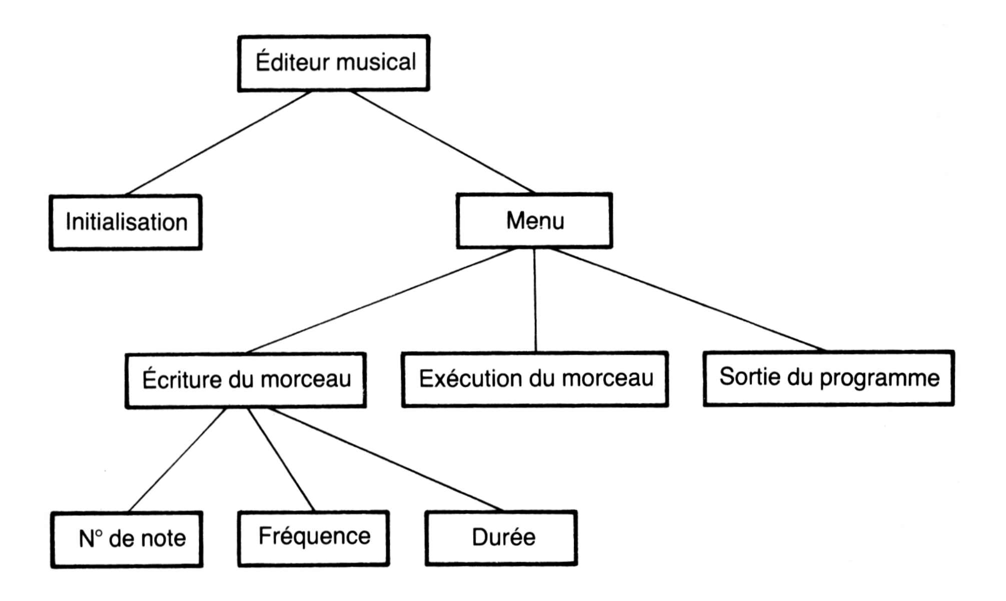
Lignes 10 à 80	: Programme principal.
Lignes 1000 à 1190	: Menu.
Ligne 1110	: Choix du morceau.
Lignes 2000 à 2200	: Exécution du morceau.
Lignes 3000 à 3090	: Pop Corn.
Lignes 3100 à 3190	: Histoires sans parole.
Lignes 3200 à 3280	: Classique.
Lignes 3300 à 3370	: Midnight Express.
Lignes 3400 à 3440	: La truite de Schubert.

ÉDITEUR MUSICAL

Si vous ne possédez pas la partition du morceau que vous voulez reproduire, mais si vous avez un tant soit peu l'oreille musicale, ce programme vous concerne. Il permet de "mettre au point" la partition que vous tentez d'immortaliser.

Une fois votre œuvre codée, il sera très facile de passer au niveau supérieur : l'écriture d'un programme du type précédent.

La structure du programme est la suivante :



Programme

```
10 REM *********************************
11 REM *                               *
12 REM *  E D I T E U R   M U S I C A L*
13 REM *                               *
14 REM *********************************
15 :
1000 REM Menu Principal
1010 :
1015 DIM NO$(300) '300 Notes au maximum
1016 :
1020 CLS:SCREEN 6,0,0:PRINT"EDITEUR MUSICAL":
PRINT
1030 PRINT"1)Modification du Tempo,"
1040 PRINT"2)Modification de l'attaque,"
1050 PRINT"3)Ecriture du morceau,"
1060 PRINT"4)Execution du morceau,"
1070 PRINT"5)Sauvegarde K7,"
1075 PRINT"6)Sortie de l'editeur."
1080 PRINT:INPUT"Votre choix (1..6) ";C
1090 IF C>6 OR C<1 THEN 1000 'Reponse refusee
1095 IF C=6 THEN 1125
1100 :
1110 ON C GOSUB 10000,11000,12000,13000,14000
1115 GOTO 1020
1120 :
1125 END
1130 REM *******************************
10000 REM Modification du Tempo
10001 :
10010 CLS:PRINT"Modification du Tempo":PRINT
10020 INPUT"Tempo (1..255) ";TP
10030 IF TP>255 OR TP<1 THEN 10000
10040 X$="T"+RIGHT$(STR$(TP),LEN(STR$(TP))-1)
10050 PLAY X$
10060 :
10070 RETURN
10080 REM ******************************
11000 REM Modification de l'Attaque
11001 :
11010 CLS:PRINT"Modification de l'Attaque":PR
INT
11020 INPUT"Attaque (0..255) ";AT
11030 IF AT>255 OR AT<0 THEN 11000
11040 X$="A"+RIGHT$(STR$(TP),LEN(STR$(TP))-1)
11050 PLAY X$
11060 :
11070 RETURN
11080 REM ******************************
```

```
12000 REM Ecriture du morceau
12001 :
12010 CLS:PRINT"Ecriture du morceau":PRINT
12020 PRINT"Notes : de DO1 a SI5, ou 0 Pour s
ortir":PRINT
12030 PRINT"Duree : 1-Croche  2-Noire"
12040 PRINT"          3-Blanche 4-Ronde"
12050 PRINT:INPUT"Note ";N$
12051 IF N$="0" THEN 12185 'Fin
12080 :
12090 PRINT:INPUT"Duree ";D
12100 IF D>4 OR D<1 THEN PLAY"L24O4DO":GOTO 1
2090
12105 GOSUB 20000 'Decodage note
12106 IF D=1 THEN PLAY"L24O4DO":GOTO 12050
12110 :
12120 PRINT:INPUT"TemPs ";T
12130 IF T>300 OR T<1 THEN PLAY"L24O4DO":GOTO
 12120
12140 :
12150 NO$(T)=ND$ 'Memorisation
12160 :
12170 GOTO 12000
12180 :
12185 RETURN
12190 REM ******************************
13000 REM Execution du morceau
13001 :
13010 I=0
13020 I=I+1
13030 IF NO$(I)="" THEN 13070 'Fin du morceau
13035 PLAY NO$(I)
13040 :
13050 GOTO 13020
13060 :
13070 RETURN
13080 REM ******************************
14000 REM Sauve9arde cassette
14001 :
14010 CLS:PRINT"Sauve9arde Cassette":PRINT
14020 INPUT"Nom de la sauve9arde ";N$
14030 :
14040 I=0
14050 I=I+1
14060 IF NO$(I)<>"" THEN 14050
14070 :
14080 OPEN"O",#1,N$
14090 PRINT#1,I 'Nombre de donnees
14100 FOR J=1 TO I
14110   PRINT#1,NO$(J)
```

```
14120 NEXT J
14130 CLOSE#1
14140 :
14150 RETURN
14160 REM ********************************
20000 REM Decodage de la note tapee
20001 :
20005 B=0 'Initialisation
20006 :
20010 L$=LEFT$(N$,LEN(N$)-1) 'Note
20020 R=ASC(RIGHT$(N$,1))-48:R$=RIGHT$(STR$(R
),1) 'Octave
20030 IF R>5 OR R<1 THEN B=1:GOTO 20070
20035 D=D*20:D$=STR$(D):D$=RIGHT$(D$,LEN(D$)-
1)
20040 :
20050 ND$="O"+R$+"L"+D$+L$
20060 :
20070 RETURN
```

Analyse du programme

Lignes 1000 à 1125	: Menu.
Lignes 10000 à 10070	: Modification du tempo.
Lignes 11000 à 11070	: Modification de l'attaque.
Lignes 12000 à 12190	: Écriture du morceau.
Lignes 13000 à 13070	: Exécution du morceau.
Lignes 14000 à 14150	: Sauvegarde cassette.
Lignes 20000 à 20070	: Décodage de la note tapée.

LECTURE DE FICHIERS MUSICAUX

Après avoir sauvegardé vos œuvres (créées par le programme précédent), il peut être intéressant de pouvoir les réécouter.

Ce programme lit un fichier musical stocké par l'"Éditeur musical" et joue le morceau ainsi constitué.

Programme

```
1000 REM ********************************
1001 REM *                              *
1002 REM * LECTURE DE FICHIERS MUSICAUX *
1003 REM *                              *
1004 REM ********************************
```

```
1005 :
1010 CLS:PRINT"   LECTURE DE FICHIERS MUSICAU
X"
1020 LOCATE 0,10:INPUT"Nom du fichier ";N$
1030 :
1040 OPEN "I",#1,N$
1050 INPUT#1,NB 'Nombre de donnees
1060 DIM A$(NB)
1070 :
1080 FOR I=1 TO NB
1090   INPUT#1,A$(I)
1100 NEXT I
1110 CLOSE#1
1120 :
1130 CLS:PRINT"Appuyez sur une touche pour ex
ecuter"
1140 PRINT"le morceau."
1150 A$=INKEY$:IF A$="" THEN 1150
1160 :
1170 REM Execution du morceau
1180 :
1190 I=0
1200 I=I+1
1210 IF A$(I)="" THEN 1260
1220 :
1230 PLAY A$(I)
1240 GOTO 1200
1250 :
1260 END
```

Analyse du programme

Ligne 1050	: Lecture du nombre de données contenues dans le fichier.
Lignes 1080 à 1100	: Lecture des notes.
Lignes 1190 à 1240	: Exécution du morceau.

Remarque : l'attaque et le tempo ne sont pas enregistrés par le programme précédent. Si vous souhaitez le faire, les modifications à apporter sont minimes.

Les deux programmes qui suivent ne sont pas à proprement parler des programmes musicaux. Le premier reproduit les battements d'un métronome, et le second montre les effets spéciaux que l'on peut créer en BASIC.

MÉTRONOME

Cet instrument, bien connu des musiciens, permet aux débutants comme aux professionnels de respecter un tempo donné.

Le programme demande le nombre de notes émises par minute (c'est le tempo) entre 10 et 300. L'appui sur une touche provoque le démarrage du métronome. La sortie du programme se fait en appuyant simultanément sur les touches "CNT" et "C".

Programme

```
1000 REM ********************************
1001 REM *                              *
1002 REM *       M E T R O N O M E      *
1003 REM *                              *
1004 REM ********************************
1005 :
1010 CLS:PRINT"                    METRONOME":LOCATE
 8,10
1020 INPUT"Tempo (10..300) ";T
1030 LOCATE 0,20,0:PRINT"Tapez sur une touche
 pour commencer."
1050 :
1060 A$=INKEY$:IF A$="" THEN 1060 'Attente ac
tion clavier
1070 :
1080 PLAY "O2L10DO"
1090 FOR I=1 TO 475/(T/60):NEXT I
1100 GOTO 1080
```

Analyse du programme

Ligne 1020	: Saisie du tempo.
Ligne 1060	: Attente de l'appui sur une touche.
Lignes 1080 à 1100	: Métronome.

EFFETS SPÉCIAUX

Ce programme montre comment réaliser des bruits d'animation du type "Jeux d'arcade" en utilisant des boucles sur l'ordre PLAY.

Programme

```
1  REM Boucles sur PLAY
2  :
10 REM Bonus 1
11 :
12 PLAY"L2"
13 FOR I=1 TO 10
14   PLAY"O3DOMISOO4DOMISOO5DOMISO"
15 NEXT I
16 REM ******************************
30 REM Bonus 2
31 :
32 PLAY "L3"
33 FOR I=1 TO 7
34   PLAY"O3DOREMIFASOLASIO4DOREMIFASOLASIO5D
OREMIFASOLASI"
35 NEXT I
36 REM ******************************
100 REM Bonus 3
101 :
102 PLAY "L5"
103 FOR I=1 TO 20
104   PLAY"O4DOO5DO"
105 NEXT I
106 REM ******************************
130 REM Bateau
131 :
132 PLAY "L2"
133 FOR I=1 TO 100
134   PLAY"O1DOMI"
135 NEXT I
136 REM ******************************
170 REM Pistolet Mitrailleur
171 :
172 PLAY "L2T1"
173 FOR I=1 TO 20
174   PLAY"O5SILASOFAMIREDOREMIFASOLASI"
175 NEXT I
```

Analyse du programme

Lignes 10 à 15	: Bonus 1.
Lignes 30 à 35	: Bonus 2.
Lignes 100 à 105	: Bonus 3.
Lignes 130 à 135	: Bateau.
Lignes 170 à 175	: Pistolet mitrailleur.

Chapitre 5 — Les modes graphiques haute résolution

Avant de présenter le premier utilitaire, décrivons la structure de la mémoire d'écran graphique. MO5 possède un mode haute résolution graphique. Sa résolution est de 320 points horizontaux sur 200 points verticaux, soit 64000 points.

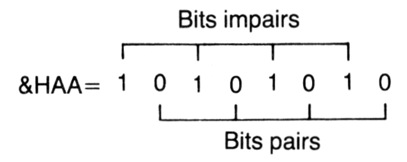
L'écran graphique se traduit en une image mémoire à partir de l'adresse 0 sur une longueur de 8000 octets ($1F40 en hexa).

Cet écran graphique permet d'afficher des couleurs différentes. Les 8000 octets ne suffisent pas à stocker les informations de forme et de couleur. Les mémoires d'écran motif et couleur occupent les mêmes adresses mémoire (0 à 8000). L'adresse $A7C0 sélectionne la mémoire d'écran motif si son bit 0 est forcé à 1, et la mémoire d'écran couleur si son bit 0 est forcé à 0.

Le codage mémoire du motif est le suivant :

	Colonne 0		Colonne 39
Ligne 0	0		39
	40		79
	7 920		7 959
Ligne 199	7 960		7 999

Chaque octet de cette mémoire renseigne l'état de 8 points élémentaires. Par exemple, si l'on fait POKE &HA7C0,PEEK(&HA7C0)OR 1:POKE 4000,&HAA.



Les 4 bits pairs auront la couleur du motif car codés 1.

Les 4 bits impairs auront la couleur du fond car codés 0.

Les couleurs motif et fond sont définies par :

POKE &HA7C0,PEEK(&HA7C0) AND 254 : POKE 4000, COULEUR

Le codage mémoire de la couleur est identique au codage motif :

l'octet 0 concerne la couleur des 8 points en haut à gauche de l'écran ;
l'octet 39 concerne la couleur des 8 points en haut à droite de l'écran.

L'octet 7999 concerne la couleur des 8 points en bas à droite de l'écran. Chaque octet de la mémoire de couleur renseigne :

- pour le quartet de poids forts, la couleur de l'encre ;
- pour le quartet de poids faibles, la couleur du fond ;

avec le code des couleurs suivant :

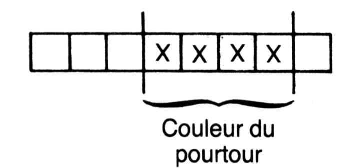
0	Noir	8	Gris
1	Rouge	9	Rouge clair
2	Vert	10	Vert clair
3	Jaune	11	Jaune clair
4	Bleu	12	Bleu clair
5	Magenta	13	Magenta clair
6	Cyan	13	Magenta clair
7	Blanc	14	Cyan clair
		15	Orange

Par exemple :

POKE &HA7C0,PEEK(&HA7C0) AND 254 : POKE 4000,&H5F

définira la couleur d'encre magenta et la couleur de fond orange pour le groupe des 8 pixels situés en adresse 4000.

Pour finir avec le codage des couleurs, les bits 1, 2, 3 et 4 de l'adresse mémoire &HA7CO donnent la couleur du pourtour de l'écran, selon le code des couleurs donné ci-dessus.



Par exemple :

POKE &HA7C0,&H0C

donnera la couleur bleu clair au pourtour.

Les BASIC MO5 et TO7/70 possèdent des ordres graphiques évolués :

- PSET permet d'allumer un point élémentaire sur l'écran ;
- LINE permet de tracer un segment de droite sur l'écran.

Nous allons étudier l'utilisation de ces deux ordres graphiques dans la suite.

TRACÉ DE COURBES EN HAUTE RÉSOLUTION

Le lycéen de second cycle est souvent confronté au problème suivant : étudier puis représenter l'évolution d'une courbe monovariable entre deux bornes fixes.

Le programme décrit ici permet de donner le tracé d'une courbe monovariable quelconque de type Y = F(x) entre deux bornes fixées par l'utilisateur.

Si ce programme ne réduit pas à néant l'étude préalable de la fonction (en particulier l'étude des points singuliers), il permet cependant de confirmer ou d'infirmer l'allure générale de la courbe à étudier.

Le programme est très simple. Il se décompose en trois phases :

- saisie de l'équation ;
- saisie du domaine de définition ;
- passage en mode haute résolution et tracé.

Programme

```
10 REM ****************************************
11 REM *                                      *
12 REM * TRACE DE COURBES D'EQUATION Y=F(X)   *
13 REM *                                      *
14 REM ****************************************
15 :
20 GOSUB 1000 'Saisie de l'equation
30 STOP
40 GOSUB 2000 'Saisie du domaine
```

```
50 GOSUB 3000 'Trace
60 :
70 END
80 REM *************************************
1000 REM Saisie de l'equation
1010 :
1020 CLS
1030 PRINT "Tapez 4020 A="
1040 PRINT "suivi de la fonction a etudier."
1050 PRINT:PRINT "Tapez ensuite RUN 40"
1060 :
1070 RETURN
1080 REM *************************************
2000 REM Saisie du domaine de definition
2010 :
2020 CLS
2030 PRINT "Entrez le domaine d'etude :"
2040 PRINT:INPUT "X min";X1
2050 PRINT:INPUT "X max";X2
2070 :
2080 RETURN
2090 REM *************************************
3000 REM Trace de la courbe
3010 :
3020 REM Calcul de l'echelle en Y
3030 :
3035 M1=-1E33:M2=1E33
3040 FOR X=X1 TO X2 STEP (X2-X1)/100
3050   GOSUB 4000 'Calcul de la fonction
3060   IF A>M1 THEN M1=A
3070   IF A<M2 THEN M2=A
3080 NEXT X
3090 EX=320/(X2-X1):EY=200/(M1-M2)
3100 PX=(X2-X1)/100
3110 :
3120 REM Trace
3130 :
3140 CLS 'Effacement d'ecran
3150 FOR X=X1 TO X2 STEP PX
3160   GOSUB 4000 'Calcul de la fonction
3170   PSET ((X-X1)*EX,200-(A-M2)*EY)
3180 NEXT X
3190 :
3200 RETURN
3210 REM *************************************
4000 REM Fonction a etudier
4010 :
4020 A=COS(X)
4030 RETURN
```

Analyse du programme

Lignes 20 à 50	: Programme principal.
Lignes 1000 à 1070	: Saisie de l'équation.
Lignes 2000 à 2080	: Saisie du domaine d'étude.
Lignes 3000 à 3100	: Calcul d'échelle.
Lignes 3140 à 3180	: Tracé de la courbe.
Lignes 4000 à 4030	: Fonction à étudier.
Ligne 4020	: Définition de la fonction à étudier.

Tracé de courbes Y(t), X(t)

Ce programme peut être employé dans deux voies différentes :

- comme aide mathématique (représentation instantanée d'une courbe par l'entrée de ses équations);
- comme générateur de motifs mathématiques graphiques.

La première voie, si elle n'évite pas les fastidieuses recherches de tangentes, points d'inflexion et assymptotes nécessaires à tout bon tracé de courbes, permet cependant de concrétiser rapidement une impression plus ou moins juste sur l'allure de la courbe à étudier.

La seconde voie permet d'obtenir d'assez jolis dessins par la seule entrée d'équations (dont quelques exemples sont donnés par la suite) et du domaine d'étude.

Programme

```
10 REM ***************************************
11 REM *                                     *
12 REM * TRACE DE COURBES D'EQUATION Y(t),X(t) *
13 REM *                                     *
14 REM ***************************************
15 :
20 GOSUB 1000 'Saisie des equations
30 STOP
40 GOSUB 2000 'Saisie du domaine
50 GOSUB 3000 'Trace
60 :
70 END
80 REM *************************************
1000 REM Saisie des equations
1010 :
1020 CLS
1030 PRINT "Tapez 4020 A="
1031 PRINT "Tapez 4021 B="
1040 PRINT "suivi des equations X(t),Y(t) a etud
ier"
1050 PRINT:PRINT "Tapez ensuite RUN 40"
1060 :
1070 RETURN
1080 REM *************************************
2000 REM Saisie du domaine de definition
2010 :
2020 CLS
2030 PRINT "Entrez le domaine d'etude :"
2040 PRINT:INPUT "T min";T1
2050 PRINT:INPUT "T max";T2
2070 :
```

```
2080 RETURN
2090 REM ***********************************
3000 REM Trace de la courbe
3010 :
3020 TPAS=(T2-T1)/320 ''Pas en X
3030 :
3040 REM Pas en X et en Y
3050 :
3060 X1=1E33:X2=-1E33:Y1=1E33:Y2=-1E33
3070 :
3080 FOR T=T1 TO T2 STEP TPAS
3090   GOSUB 4000 'Calcul des fonctions
3100   IF A<X1 THEN X1=A:X3=T
3110   IF A>X2 THEN X2=A:X4=T
3120   IF B<Y1 THEN Y1=B:Y3=T
3130   IF B>Y2 THEN Y2=B:Y4=T
3140 NEXT T
3150 :
3160 XPAS=(X1-X2)/320:YPAS=(Y1-Y2)/200
3170 :
3180 REM Trace
3190 :
3200 CLS 'Effacement d'ecran
3210 FOR T=T1 TO T2 STEP TPAS
3220   GOSUB 4000 'Calcul des fonctions
3230   PSET(INT((1/XPAS)*(X1-A)),200-INT((1/YPAS
)*(B-Y2))),6
3240 NEXT T
3250 :
3260 RETURN
4000 REM Fonctions a etudier
4010 :
4020 A=SIN(T)
4021 B=COS(T)
4030 :
4040 RETURN
```

Analyse du programme

Le programme possède la même structure que le précédent.

Tracé de courbes Y(t),X(t) imbriquées

Ce programme s'adresse à tous ceux que les belles représentations mathématiques font rêver.

C'est une variante du programme précédent. Il permet d'imbriquer autant

de courbes de même équation qu'on le désire, ce qui donne une profondeur au tracé.

Seulement, quelques lignes ont été changées par rapport à “Tracé de courbes Y(t),X(t)” :

- dans la saisie du domaine d'étude,
- dans le tracé de la courbe.

Programme

```
10 REM ****************************************
11 REM *                                      *
12 REM * TRACE DE COURBES D'EQUATION Y(t),X(t) *
13 REM *                                      *
14 REM ****************************************
15 :
20 GOSUB 1000 'Saisie des equations
30 STOP
40 GOSUB 2000 'Saisie du domaine
50 GOSUB 3000 'Trace
60 :
70 END
80 REM **************************************
1000 REM Saisie des equations
1010 :
1020 CLS
1030 PRINT "Tapez 4020 A="
1031 PRINT "Tapez 4021 B="
1040 PRINT "suivi des equations X(t),Y(t) a etud
ier"
1050 PRINT:PRINT "Tapez ensuite RUN 40"
1060 :
1070 RETURN
1080 REM **************************************
2000 REM Saisie du domaine de definition
2010 :
2020 CLS
2030 PRINT "Entrez le domaine d'etude :"
2040 PRINT:INPUT "T min";T1
2050 PRINT:INPUT "T max";T2
2060 PRINT:INPUT "Nombre de courbes imbriquees";
NC
2061 PRINT:INPUT "Nombre de Points Par courbe ";
NP
2070 :
2080 RETURN
2090 REM **************************************
3000 REM Trace de la courbe
3010 :
3020 TPAS=(T2-T1)/NP 'Pas en T
3030 :
3040 REM Pas en X et en Y
```

```
3050 :
3060 X1=1E33:X2=-1E33:Y1=1E33:Y2=-1E33
3070 :
3080 FOR T=T1 TO T2 STEP TPAS
3090   GOSUB 4000 'Calcul des fonctions
3091   A=A*NC:B=B*NC
3100   IF A<X1 THEN X1=A:X3=T
3110   IF A>X2 THEN X2=A:X4=T
3120   IF B<Y1 THEN Y1=B:Y3=T
3130   IF B>Y2 THEN Y2=B:Y4=T
3140 NEXT T
3150 :
3160 XPAS=(X1-X2)/320:YPAS=(Y1-Y2)/200
3170 :
3180 REM Trace
3190 :
3200 CLS 'Effacement d'ecran
3205 FOR K=1 TO NC
3210   FOR T=T1 TO T2 STEP TPAS
3220     GOSUB 4000 'Calcul des fonctions
3221     A=A*K:B=B*K
3230     PSET(INT((1/XPAS)*(X1-A)),200-INT((1/YP
AS)*(B-Y2))),6
3240   NEXT T
3245 NEXT K
3250 :
3260 RETURN
3270 REM ************************************
4000 REM Fonctions a etudier
4010 :
4020 A=SIN(T)
4021 B=COS(T)
4030 :
4040 RETURN
```

Analyse du programme

Le programme possède la même structure que les deux précédents.

Exemples d'équations

Essayez diverses combinaisons de fonctions trigonométriques. Les couples d'équations suivants donnent de bons résultats avec 150 points et un domaine d'étude compris entre 0 et 56.

(1+COS(T)) *SIN(T) (1+COS(T)) *COS(T)	(1−SIN(T)) *COS(T) (1+COS(T)) *COS(T)
(1+COS(2*T)) *SIN(2*T) (1+COS(T)) *SIN(T)	(1−COS(2*T)) *SIN(T) (1+COS(T)) *COS(T)
(1+SIN(2*T)) *COS(T) (1+SIN(2*t)) *SIN(T)	etc...

FIGURES DE MOIVRE

Les figures obtenues par le glissement d'une droite sur deux courbes planes portent le nom de “figures de MOIVRE”.

De très belles figures peuvent être obtenues sur les micro-ordinateurs graphiques, à moindre effort.

Les programmes proposés ici se servent de l'ordre BASIC **“LINE (X1,Y1)–(X2,Y2)”** défini quelques pages plus loin. Pour comprendre la logique de ce programme, reportez-vous à cette définition.

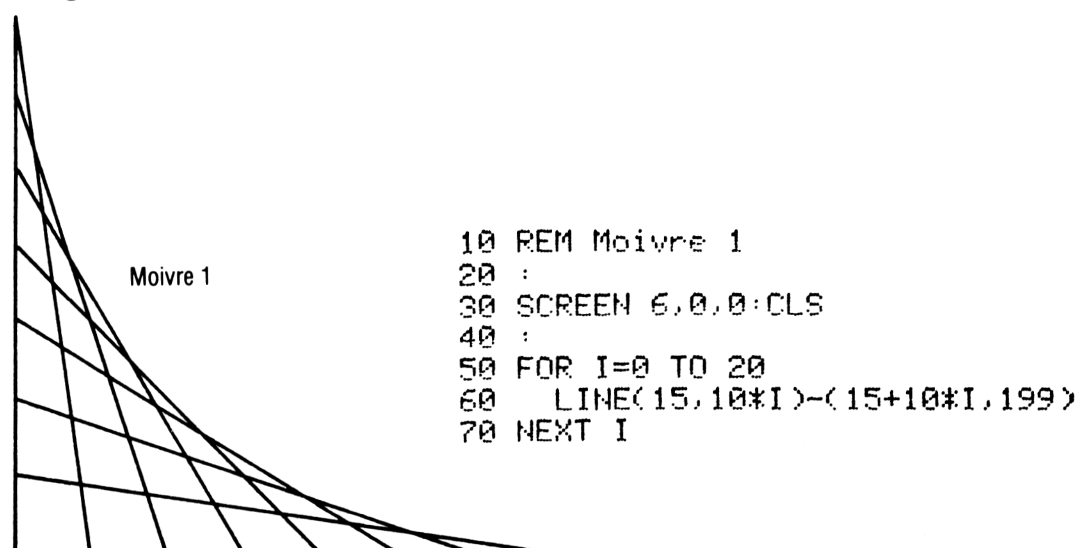
Sept figures sont proposées :

- MOIVRE 1 : les deux courbes planes sont deux droites d'angle 90°.
- MOIVRE 2 : les deux courbes planes sont deux droites d'angle < 90°.
- MOIVRE 3 : composition de quatre figures du type 1.
- MOIVRE 4 : autre composition de quatre figures de type 1.
- MOIVRE 5 : composition de quatre figures de type 2.
- MOIVRE 6 : composition de deux figures de type 2 pour former un triangle.
- MOIVRE 7 : application de “MOIVRE 1” pour former le dessin d'un papillon.

Organigramme des programmes

Chacun des sept programmes possède une structure linéaire qui consiste en une ou plusieurs boucles décrivant le glissement de la droite.

Programmes



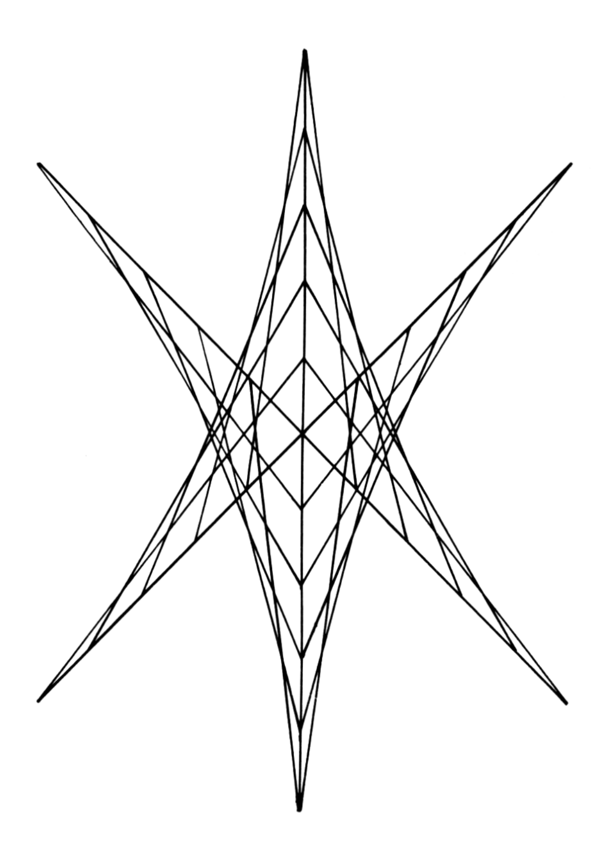

```
10 REM Moivre 1
20 :
30 SCREEN 6,0,0:CLS
40 :
50 FOR I=0 TO 20
60   LINE(15,10*I)-(15+10*I,199)
70 NEXT I
```

```
10 REM Moivre 2
20 :
30 SCREEN 6,0,0:CLS
40 :
50 FOR I=0 TO 20
60   LINE(15,10*I)-(15+10*I,199-9*I)
70 NEXT I
```

Moivre 2

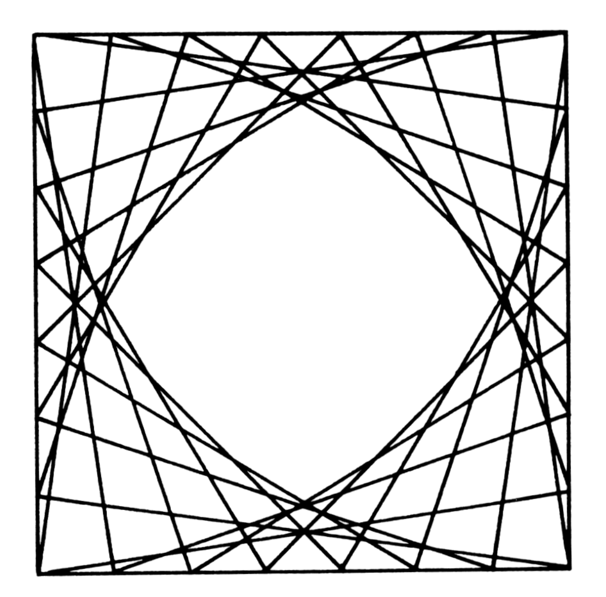
Moivre 3

```
10  REM Moivre 3
20  :
30  SCREEN 6,0,0:CLS
40  :
50  FOR I=0 TO 11
60    LINE(60,16*I)-(60+16*I,199)
70    LINE(60+I*16,0)-(260,16*I)
80    LINE(60,199-16*I)-(60+16*I,0)
90    LINE(60+16*I,199)-(260,199-16*I)
100 NEXT I
110 LINE (260,0)-(260,208)
```



Moivre 4

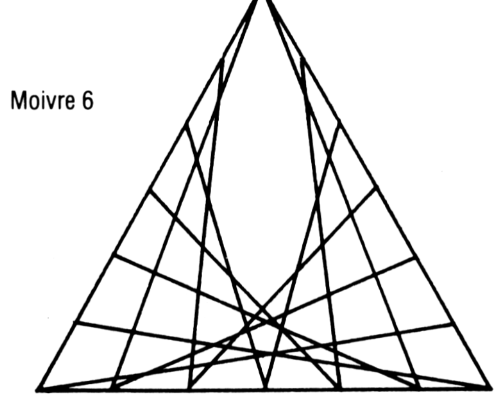
```
10  REM Moivre 4
20  :
30  SCREEN 6,0,0:CLS
40  :
50  FOR I=0 TO 11
60    LINE(60+I*9,99)-(160,99+9*I)
70    LINE(160,99+9*I)-(260-9*I,99)
80    LINE(60+9*I,99)-(160,99-9*I)
90    LINE(160,99-9*I)-(260-9*I,99)
100 NEXT I
```



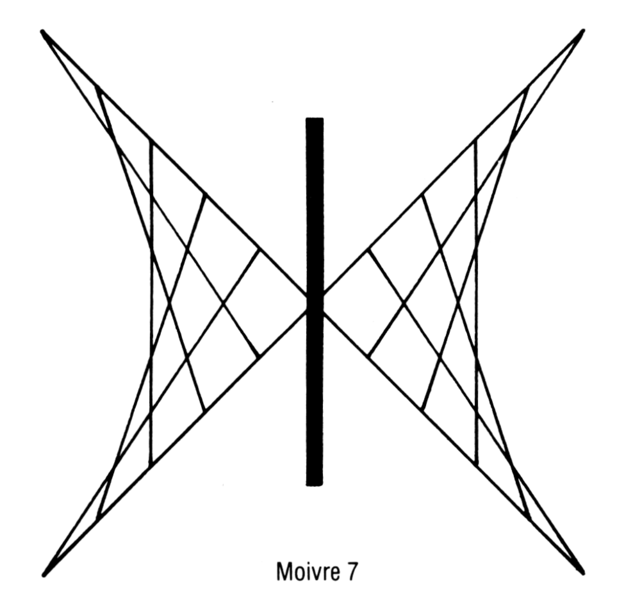
Moivre 5

```
10  REM Moivre 5
20  :
30  SCREEN 6,0,0:CLS
40  :
50  FOR I=0 TO 11
60    LINE(60+I*9,99)-(160-9*I,99+9*I)
70    LINE(60+I*9,99)-(160-9*I,99-9*I)
80    LINE(259-I*9,99)-(160+9*I,99-9*I)
90    LINE(259-I*9,99)-(160+9*I,99+9*I)
100 NEXT I
```

Moivre 6



```
10  REM Moivre 6
20  :
30  SCREEN 6,0,0:CLS
40  :
50  FOR I=1 TO 10
60    LINE(110+I*10,150)-(165-I*5,30+I*10)
70    LINE(210-I*10,150)-(155+I*5,30+I*10)
80  NEXT I
90  LINE(105,150)-(215,150)
100 LINE(215,150)-(160,40)
110 LINE(160,40)-(105,150)
```



Moivre 7

```
10  REM Moivre 7, Le PaP
20  :
30  SCREEN 6,0,0:CLS
40  :
50  FOR I=0 TO 10
60    LINE(I*4+120,140-I*4)-(160-I*4,100-I*4)
70    LINE(200-I*4,140-I*4)-(160+I*4,100-I*4)
80  NEXT I
90  BOX (159,70)-(161,130)
100 BOXF(159,70)-(161,130)
```

SIMULATION D'ORDRES GRAPHIQUES ÉVOLUÉS

La plupart des micro-ordinateurs possédant un graphisme haute résolution disposent souvent de l'ordre **CIRCLE** pour tracer des cercles et parfois de l'ordre **ARC** pour tracer des arcs de cercles.

Le programme qui suit permet de simuler l'ordre CIRCLE. Le tracé est fait en BASIC. Un "certain temps" est donc nécessaire pour tracer un cercle ou un arc de cercle.

CIRCLE

Ce programme trace un cercle si on lui fournit les coordonnées du centre du cercle et le rayon du cercle. Il se sert du fait que l'équation Y(t), X(t) d'un cercle est de la forme :

$$X = X1 + R\ COS\ (t)$$
$$Y = Y1 + R\ SIN\ (t)$$

avec X1,Y1 coordonnées du centre du cercle et R rayon du cercle.

Il suffit alors de faire varier t de 0 à 2Π pour obtenir un tracé complet du cercle.

Exemple

```
10 CLS:X1=100:Y1=100:R=50:AD=270:AF=360:GOSUB 10
000
20 END
```

Programme

```
10000 REM ********************************
10001 REM *                              *
10002 REM *TRACE   D'ARCS   DE   CERCLES *
10003 REM *                              *
10004 REM ********************************
10005 REM *                              *
10006 REM * Entree :X1, Y1 Coord. Centre *
10007 REM *         R  Rayon du cercle   *
10008 REM *         AD Angle de depart   *
10009 REM *         AF Angle de fin      *
10010 REM *                              *
10011 REM ********************************
10012 :
10030 FOR I=AD TO AF
10040   X=X1+R*COS(I*3.14159265/180)
10050   Y=Y1-R*SIN(I*3.14159265/180)
10060   PSET(X,Y),3
10070 NEXT I
10080 :
10090 RETURN
```

Analyse du programme

Lignes 10020 à 10070 : Simulation de l'ordre CIRCLE.

Le programme suivant permet de tracer des arcs de cercles. Le tracé est fait en BASIC.

ARC

Ce programme trace un arc de cercle si on lui fournit les coordonnées du centre du cercle, le rayon du cercle, les angles de départ et de fin de tracé (en degrés).

Exemple

```
10000 REM ************************************
10001 REM *                                  *
10002 REM *   TRACE   D'ARCS   DE   CERCLES  *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : X1, Y1 Coord. Centre    *
10007 REM *          R  Rayon du cercle      *
10008 REM *          AD Angle de depart (degre) *
10009 REM *          AF Angle de fin (degre) *
10010 REM *                                  *
10011 REM ************************************
10012 :
10030 FOR I=AD TO AF
10040   X=X1+R*COS(I*3.14159265/180)
10050   Y=Y1-R*SIN(I*3.14159265/180)
10060   PSET(X,Y),3
10070 NEXT I
10080 :
10090 RETURN
```

Programme

```
10 CLS:X1=100:Y1=100:R=50:AD=270:AF=360:GOSUB 1000
0
20 END
```

Analyse du programme

Ligne 10040 : Calcul abscisse.
Ligne 10050 : Calcul ordonnée.
Ligne 10060 : Affichage d'un point élémentaire.

PALETTE DE COULEURS

Si l'on rapproche petit à petit deux points élémentaires de couleurs différentes, il arrive un moment où l'œil ne fait plus la différence entre les deux points : il perçoit un seul point dont la couleur est le mélange de la couleur de chaque point.

Ce principe est utilisé ici : la superposition d'un fond de couleur A et d'un motif de couleur B arrive à tromper l'œil qui perçoit une couleur C différente de A et de B si le motif est bien choisi.

T07 possède 8 couleurs. MO5 possède 16 couleurs. Ce programme vous propose d'étendre le nombre de couleurs à :

- 64 pour T07,
- 256 pour MO5.

Programme

```
10  REM Palette de couleurs MO5
20  :
30  CLEAR,,1 'Declaration d'un motif graphique
40  GOSUB 1000 'Definition du motif graphique
50  :
60  FOR I=0 TO 15
70    GOSUB 2000 'Affichage d'une serie de Paste
ls
80  NEXT I
90  LOCATE 1,1,1 'Curseur apparent
100 CLS:END
110 REM ***********************************
1000 REM Definition du motif graphique
1010 :
1020 SCREEN 6,0,0
1030 DEFGR$(0)=85,170,85,170,85,170,85,170
1040 :
1050 RETURN
1060 REM ***********************************
2000 REM Affichage d'une serie de Pastels
2010 :
2020 CLS :ATTRB 1,1:COLOR 1
2030 PRINT:PRINT"PALETTE DE COULEURS"
2040 :
2050 ATTRB 0,0:LOCATE 0,5
2055 PRINT"Trame  Fond":PRINT
2060 :
2070 FOR J=0 TO 15
2075   LOCATE 2,J+6
```

```
2080   COLOR J,0:PRINT GR$(0);
2090   COLOR 6,0:PRINT"  +  ";
2100   COLOR J,I:PRINT" ";
2110   COLOR 6,0:PRINT"  =  ";
2120   COLOR J,I:PRINT GR$(0);
2130   COLOR 6,0:PRINT"    : COLOR"J","I
2140 NEXT J
2150 :
2160 PRINT:PRINT"Appuyez sur une touche pour con
tinuer."
2170 LOCATE 1,1,0 'Le curseur disparait
2180 :
2190 A$=INPUT$(1) 'Attente de l'appui
2200 :
2210 RETURN
```

Analyse du programme

Lignes 10 à 90 : Définition du motif graphique et affichage.
Lignes 1000 à 1060 : Définition du motif graphique.
Lignes 2000 à 2210 : Affichage d'une série de pastels.

FONCTION TAG

Certains micro-ordinateurs familiaux permettent d'afficher les caractères standard (ou redéfinis) à partir d'une position graphique quelconque. L'utilitaire qui suit dote les MO5 et T07/70 de cette fonction.

Vous pouvez afficher un caractère défini dans le tableau D (de manière identique à DEFGR$) à une position graphique CO, LI quelconque (avec CO compris entre 0 et 319 et LI compris entre 0 et 199).

Exemple

```
10 CLS:LOCATE 8,1,0:PRINT"Demonstration de TAG"
20 D(1)=&H18:D(2)=&H38:D(3)=&H18:D(4)=&H18:D(5)=
&H18:D(6)=&H18:D(7)=&H7E:D(8)=0
30 LI=100 'Ligne 100
40 FOR CO=1 TO 30 STEP 2
50   GOSUB 11000
60 NEXT CO
70 END
```

Programme

```
11000 REM ************************************
11001 REM *                                  *
11002 REM *          FONCTION      TAG       *
11003 REM *                                  *
11004 REM ************************************
11005 REM *                                  *
11006 REM * Entree : LI, CO Lig et Col TAG   *
11007 REM *          D      Donnee graphique *
11008 REM *                                  *
11009 REM ************************************
11010 :
11015 POKE &HA7C0,PEEK(&HA7C0) OR 1
11020 AV=CO-INT(CO/8)*8 'Nb de Bits dans la 1ere
 case
11030 C=INT(CO/8) '1ere Case du TAG
11040 FOR I=1 TO 8
11050   A=(D(I)AND 255-(2^AV-1))/2^AV
11060   POKE LI*40+C+(I-1)*40,A
11070 NEXT I
11080 FOR I=1 TO 8
11090   A=(D(I)AND 2^AV-1)*2^(8-AV)
11100   POKE LI*40+C+(I-1)*40+1,A
11110 NEXT I
11120 :
11130 RETURN
```

Analyse du programme

Ligne 11015	: Sélection de la mémoire d'écran motif.
Ligne 11020	: Calcul du nombre de bits à afficher dans le 1er pavé.
Ligne 11030	: Calcul 1re case du TAG.
Lignes 11040 à 11070	: Constitution du 1er pavé.
Lignes 11080 à 11110	: Constitution du 2e pavé.

Le programme suivant est une application de la fonction TAG.

REPÈRES Oxy PARAMÉTRABLES

L'affichage des courbes planes se fait souvent dans un repère Oxy, où x indique les abscisses et y les ordonnées. Ce programme permet d'afficher sur l'écran un repère Oxy si on lui fournit les données suivantes :

- T (1) = Abscisse origine (entre 0 et 319) ;
- T (2) = Ordonnée origine (entre 0 et 199) ;
- T (3) = Dim Ox (entre 0 et 319) ;
- T (4) = Dim Oy (entre 0 et 199) ;
- T (5) = Nombre de repères sur l'axe Ox ;
- T (6) = Nombre de repères sur l'axe Oy ;
- T (7) = Valeur numérique de X sur le point origine O ;
- T (8) = Incrément de X entre deux graduations ;
- T (9) = Valeur numérique de Y sur le point origine O ;
- T (10) = Incrément de Y entre deux graduations.

Exemple

```
10 FOR I=1 TO 10:READ T(I):NEXT I
20 DATA 20,20,100,150,6,5,10,3,5,5
30 CLS:GOSUB 10000
40 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM * TRACE DE REPERES Oxy PARAMETRABLES *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : T(1) =Abcisse Origine    *
10007 REM *          T(2) =Ordonnee Origine   *
10008 REM *          T(3) =Dimension Ox       *
10009 REM *          T(4) =Dimension Oy       *
10010 REM *          T(5) =Nombre de reperes Ox*
10011 REM *          T(6) =Nombre de reperes Oy*
10012 REM *          T(7) =Val.de X a l'origine*
10013 REM *          T(8) =Increment de X     *
10014 REM *          T(9) =Val.de Y a l'origine*
10015 REM *          T(10)=Increment de Y     *
10016 REM *                                  *
10017 REM ************************************
10018 :
10020 X0=T(1):Y0=T(2):LX=T(3):LY=T(4)
10030 NX=T(5):NY=T(6):DX=T(7):IX=T(8)
10040 DY=T(9):IY=T(10) 'Initialisation
10050 :
10060 REM Trace des axes
10070 :
```

```
10080 LINE(X0,199-Y0)-(X0+LX,199-Y0)
10090 LINE(X0,199-Y0)-(X0,199-Y0-LY)
10100 :
10110 REM Trace des fleches
10120 :
10130 CO=X0+LX:LI=199-Y0-4
10140 FOR I=1 TO 8:READ D(I):NEXT I
10150 DATA &HC0,&H60,&H30,&H18,&H18,&H30,&H60,&H
C0
10160 GOSUB 11000 ' Trace 1ere fleche
10170 :
10180 CO=X0-4:LI=199-(Y0+LY)-5
10190 FOR I=1 TO 8:READ D(I):NEXT I
10200 DATA 0,0,0,&H18,&H3C,&H66,&HC3,&H81
10210 GOSUB 11000 ' Trace 2eme fleche
10220 :
10230 REM Trace des graduations
10240 :
10250 FOR I=1 TO NX
10260   A=X0+(LX/(NX+1))*I:B=199-Y0+5
10270   LINE (A,B)-(A,B-10)
10280 NEXT I
10290 :
10300 FOR I=1 TO NY
10310   A=X0-5:B=199-(Y0+(LY/(NY+1))*I)
10320   LINE (A,B)-(A+10,B)
10330 NEXT I
10340 :
10350 REM Calcul des echelles,
10360 :
10370 MX=ABS(DX+IX*NX) 'Maximum en X
10380 MY=ABS(DY+IY*NY) 'Maximum en Y
10390 :
10400 I=0
10410 IF MX<1 THEN 10490
10420 :
10430 I=I+1
10440 MX=MX/10
10450 IF MX>1 THEN 10430
10460 PX=I-1
10470 GOTO 10540
10480 :
10490 I=I+1
10500 MX=MX*10
10510 IF MX<1 THEN 10490
10520 PX=-I
10530 :
10540 I=0
10550 IF MY<1 THEN 10620
10560 :
10570 I=I+1
10580 MY=MY/10
10590 IF MY>1 THEN 10570
10600 PY=I-1
10605 GOTO 10670
10610 :
10620 I=I+1
```

```
10630 MY=MY*10
10640 IF MY<1 THEN 10620
10650 PY=-I
10660 :
10670 REM Annotations
10680 :
10690 LOCATE X0/8-2,25-(Y0/8)
10700 PRINT ((DX/10^PX)*10)/10;"+";
10710 PRINT((IX/10^PX)*10)/10;"^";
10720 PRINTPX
10730 :
10740 LOCATE 0,25-((Y0+LY)/8)-2
10750 PRINT ((DY/10^PY)*10)/10;"+";
10760 PRINT ((IY/10^PY)*10)/10;"^";
10770 PRINT PY
10780 :
10790 RETURN
11000 REM ***********************************
11001 REM *                                 *
11002 REM *        FONCTION      TAG        *
11003 REM *                                 *
11004 REM ***********************************
11005 REM *                                 *
11006 REM * Entree : LI, CO Lig et Col TAG  *
11007 REM *          D     Donnee graphique *
11008 REM *                                 *
11009 REM ***********************************
11010 :
11015 POKE &HA7C0,PEEK(&HA7C0) OR 1
11020 AV=CO-INT(CO/8)*8 'Nb de Bits dans la 1ere
 case
11030 C=INT(CO/8) '1ere Case du TAG
11040 FOR I=1 TO 8
11050   A=(D(I)AND 255-(2^AV-1))/2^AV
11060   POKE LI*40+C+(I-1)*40,A
11070 NEXT I
11080 FOR I=1 TO 8
11090   A=(D(I)AND 2^AV-1)*2^(8-AV)
11100   POKE LI*40+C+(I-1)*40+1,A
11110 NEXT I
11120 :
11130 RETURN
```

Analyse du programme

Lignes 10020 à 10040 : Conversion du tableau en variables explicites.
Lignes 10060 à 10090 : Tracé des axes.
Lignes 10110 à 10210 : Tracé des flèches en bout d'axes.
Lignes 10230 à 10330 : Tracé des graduations sur Ox et Oy.
Lignes 10350 à 10650 : Calcul des échelles sur Ox et Oy.
Lignes 10670 à 10770 : Annotations sur les axes Ox et Oy.

Pour finir ce chapitre sur le mode haute résolution, voici un programme qui vous permet d'obtenir sur imprimante le dessin d'un objet défini en basse résolution au moyen de différentes lettres.

DESSIN EN BASSE RÉSOLUTION : TITI

Le dessin choisi est celui de TITI. Vous devez posséder une imprimante 80 colonnes ou plus pour exécuter ce programme. Plusieurs lettres sont utilisées afin d'obtenir différents contrastes. Le programme consiste en l'impression du dessin ligne par ligne.

Programme

```
10 REM TITI
20 :
40 OPEN "O",#1,"LPRT:"
50 PRINT#1,SPC(36)"O"
55 PRINT#1,SPC(30)"O    O"
60 PRINT#1,SPC(26)"O   O    O"
70 PRINT#1,SPC(27)"O   O    O"
80 PRINT#1,SPC(27)"O   O    O"
90 PRINT#1,SPC(24),"BPPPPPPPPPPPPPPB"
100 PRINT#1,SPC(19)"BPPPPPPPPPPPPPPPPPPPPPPPPB"
110 PRINT#1,SPC(16)"BPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPB"
120 PRINT#1,SPC(13)"BPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPB"
130 PRINT#1,SPC(11)"BPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPB"
140 PRINT#1,SPC(9)"BPPPPBPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPBPPPB"
150 PRINT#1,SPC(8)"BPPPBPPPPBPPPPPPPPPPPPPPPPPPP
PPPPPPPPPBPPPPPBPPB"
160 PRINT#1,"       BPPBPPPBPPPPPPPPPP PPPPPPPPP
PPPPPPPPPPPPPPBPPPPBPB"
170 PRINT#1,"      BPPBPPBPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPBPPPBPB"
180 PRINT#1,"     BPPPPPBPPPPBPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPBPPPPPB"
190 PRINT#1,"    BPPPPPBPPBPBPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPBPPPPBPPPPPBP"
200 PRINT#1,"    PPPPPPPPPPPBPPBPPPPPPPPPPPPPPPP
PPPPPPPPPPPPBPBPPPPPPPPPP"
210 PRINT#1,"   BPPPPPPPPPBPBPPBPPPPPPPPPPPPPPPP
PPPPPPPPPPPPBPBPPPBPPPPPPB"
220 PRINT#1,"   BPPPPPPPPPPBPPBPPBPPPPPPPPPPPPPP
PPPPPPPPPPPPBPBPPPPPPPPPPB"
230 PRINT#1,"   BPPPPPPPPPPPBPPBPPBPPPPPPPPPPPPP
PPPPPPPPPPBPBPPPBPPPPPPPPB"
```

```
240 PRINT#1,"  BPPPPPPPPPPPBPPB   PPPPPPPPPPPPP
PPPPPPP   BPPBPPPPPPPPPPPB"
250 PRINT#1,"   PPPPPPPPPPPPB       PPPPPPPPPPP
PPPPP       BPPPPPPPPPPPP"
260 PRINT#1,"   BPPPPPPPPPPP       PPPPPPPPPPP
PPPP        PPPPPPPPPPPB"
270 PRINT#1,"    BPPPPPPPPPP        PPPPPPPPPPP
PPP        PPPPPPPPPPPPS"
280 PRINT#1,"      BPPPPPPPP         PPPPPPPPPP
PPP         PPPPPPPPPPB"
290 PRINT#1,"       BPPPPPPPP        PPPPPPPPPP
PP          PPPPPPPPPB"
300 PRINT#1,"       BPPPPPPP   OO    PPPPPPPPPP
P    OO    PPPPPPPPPB"
310 PRINT#1,"        BPPPPPP  OOOO   PPPPPPPPPP
P   OOOO   PPPPPPPB"
320 PRINT#1,"         BPPPPP  OOOOO  PPPPPPPPPP
P  OOOOO   PPPPPPB"
330 PRINT#1,"          BPPPPP  OOOOO  PPPPPPPPPP
P  OBBOO PPPPPPPB"
340 PRINT#1,"          BPPPPP  OOOOO  PPPPPPPPPP
P  OBBOO PPPPPPPB"
350 PRINT#1,"           BPPPPP OOBBO  PPPPPPPPPP
P  OBBOO PPPPPPB"
360 PRINT#1,"            BPPPP OOBBB  PPPPPPPPPP
P  BBBOO PPPPPB"
370 PRINT#1,"            BPPPPP OOBB PPPPPPPPPPP
PPBBOO PPPPPPPB"
380 PRINT#1,"           BPPPPPPPPOBPPPPPPPPPPPPP
PPBOOPPPPPPPPPPPB"
390 PRINT#1,"          BPPPPPPPPPPPPPPPPP
 PPPPPPPPPPPPPPPPB"
400 PRINT#1,"          BPPPPPPPPPPPPPPPBPP
 BPPPPPPPPPPPPPPPB"
410 PRINT#1,"          BPPPPPPPPPPPPPBPP
  PPBPPPPPPPPPPPPB"
420 PRINT#1,"            BPPPPPPPPPBPPPPP
PPPPPBPPPPPPPPPS"
430 PRINT#1,"               BPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPB"
440 PRINT#1,SPC(26)"BDPPPPPPPDB"
450 PRINT#1,SPC(27)"BPPPPPPPB"
460 PRINT#1,SPC(28)"BPPPPPB"
470 PRINT#1,SPC(29)"PPPPP"
480 PRINT#1,SPC(29)"BPPPB"
490 PRINT#1,SPC(29)"BPPPB"
500 PRINT#1,SPC(26)"BBPBPPPBB"
510 PRINT#1,SPC(25)"BPPPBPPPPBB"
520 PRINT#1,SPC(25)"BPPPPPPPPPB"
```

```
530 PRINT#1,SPC(24)"BPPPPPPPPPPBB"
540 PRINT#1,SPC(23)"BPPBPPPPPPPPBPB"
550 PRINT#1,SPC(22)"BPPPBPPPPPPPPPBPPB"
560 PRINT#1,SPC(21)"BPPPPPPPPPPPPPPBPPPB"
570 PRINT#1,SPC(20)"BPPPPBPPPPPPPPPPBBPPB"
580 PRINT#1,SPC(21)"BPPPPBPPPPPPPPPPPPBPB"
590 PRINT#1,SPC(21)"BPPPBPPPPPPPPPPPPPPB"
600 PRINT#1,SPC(21)"BPPBPPPPPPPPPPPPPPPPB"
610 PRINT#1,SPC(22)"BBPPPPPPPPPPPPPPPPPB"
620 PRINT#1,SPC(22)"BPPPPPPPPPPPPPPPPPPB"
630 PRINT#1,SPC(21)"BBPPPPPPPPPPPPPPPPPPB"
640 PRINT#1,SPC(20)"BBPPPPPPPPPPPPPPPPPPB"
650 PRINT#1,SPC(20)"BBPPPPPPPPPPPPPPPPPB"
660 PRINT#1,SPC(20)"BBPPPPPPPPPPPPPPPPB"
670 PRINT#1,SPC(20)"BBPPPPPPPPPPPPPPB"
680 PRINT#1,SPC(20)"BBBPPPPPPPPPPBPPPB"
690 PRINT#1,SPC(22)"BBPPPPB    BPPB"
700 PRINT#1,SPC(25)"BPPB    BPPPB          BBB
  BBB"
710 PRINT#1,"     BBB               BPPB   BPPBP
PPBBBPPPPPPPPPPPPPPPPPPBB"
720 PRINT#1,"   BPPPPPPPBB           BPPPPPBPPPP
PPPPPPPPPPPPPPPPPPPPPPBPPPB"
730 PRINT#1," BPPPBPPPPPPPPPPBBBBBBPPPPPPPPPBBPP
PPPPPPPPPPPPPPPPPPPPPPPPPBPB"
740 PRINT#1,"BPPBPPPPPPPPPPPPPPPPPPPPPPPPPPPPB
BBPPPPPPPPPPPPPPPPPPPPPPPS"
750 PRINT#1,"BPBPPPPPPPPPPPPPPPPPPPPPPPPPPPB
    B  PPPPPPPPPPPPPPPPPPPPPPB"
760 PRINT#1," BPPPPPPPPPPPPPPPPPPPPPPPPPPPB
         BBPPPPPPPPPPPPPPPPB"
770 PRINT#1,"BPPPPPPPPPPPPPPPPPPPPPPPPB
              BBBBBBB"
780 PRINT#1,"  BPPPPPPPPPPPPPPPPPPB"
790 PRINT#1,"     BBBBBBBBBB"
```

Exécution du programme

Voir page ci-contre

Si vous désirez faire d'autres programmes du même type, reproduisez sur papier l'objet désiré. Quadrillez le papier selon la largeur et la hauteur des caractères délivrés sur votre imprimante. Utilisez les lettres foncées pour définir les contours (B,R,A), et les lettres claires pour le dessin (C,P,D,I,O).

Je vous souhaite beaucoup de courage car l'opération est longue et fastidieuse. Le résultat en vaut cependant la peine...

Chapitre 6 — Animation graphique

Ce chapitre donne au lecteur le moyen de constituer en BASIC des jeux dans lesquels un ou plusieurs objets sont animés.

Il est constitué de 4 parties :
1. - définition de caractères graphiques,
2. - utilisation des caractères graphiques,
3. - dessin sur l'écran avec le crayon optique,
4. - lecture et sauvegarde d'écrans graphiques.

DÉFINITION DE CARACTÈRES GRAPHIQUES

Les caractères des MO5 et TO7/70 sont inscrits dans une grille 8×8 pixels. 128 caractères au maximum peuvent être redéfinis par le programmeur.

Pour redéfinir N caractères graphiques, employer les instructions :

```
CLEAR,,N
DEFGR$(I)=L1 ... L8 pour I variant de 1 à N.
```

Pour afficher le caractère redéfini I, employer l'instruction PRINT GR$(I) ou encore PRINT CHR$(128+I).

Le programme suivant est un éditeur de caractères graphiques simples. Une grille 8×8 est affichée à l'écran. Vous pouvez vous déplacer dans cette grille grâce aux touches flèches, allumer un point avec P, éteindre un point avec V. Le caractère défini, appuyez sur ENTER. La suite des codes à définir dans le DEFGR$ apparaît à l'écran.

Programme

```
10 REM Programmation de caracteres graphiques
11 :
20 CLS:DIM T(8,8)
30 FOR I=1 TO 8
40   PRINT "........"
50 NEXT I
60 X=0:Y=0 'Initialisation
70 LOCATE X,Y
80 A$=INKEY$:IF A$="" THEN 80
85 A=ASC(A$)
90 IF A$="P" THEN T(Y,X)=1:PRINT"*"
100 IF A$="V" THEN T(Y,X)=0:PRINT"."
110 IF A=9  AND X<7 THEN X=X+1
120 IF A=8  AND X>0 THEN X=X-1
130 IF A=10 AND Y<7 THEN Y=Y+1
140 IF A=11 AND Y>0 THEN Y=Y-1
150 IF A<>13 THEN 70
160 LOCATE 1,15:PRINT"Donnees correspondantes:":
PRINT
170 FOR I=0 TO 7
180   A=0 'Initialisation
190   FOR J=0 TO 7
200     A=A+(2^(7-J))*T(I,J)
210   NEXT J
220   PRINT A;
230 NEXT I
```

Analyse du programme

Lignes 30 à 50	: Affichage de la grille d'écran.
Ligne 80	: Attente d'une action au clavier.
Lignes 90 à 150	: Action en fonction de la touche pressée.
Ligne 160	: Affichage des DATA du DEFGR$.

La définition d'un objet mettant en œuvre plusieurs caractères redéfinis est complexe si l'on se contente de l'utilitaire précédent.

Le programme qui suit permet de saisir un objet graphique comportant 10 caractères au maximum de large sur 6 caractères au maximum de haut.

DÉFINITION DE CARACTÈRES GRAPHIQUES MULTIPLES

Ce programme utilise 15 caractères graphiques appelés GR$(0) à GR$(14) qui permettent de multiplier par 2 le nombre de caractères affichables à l'écran, soit 80 en largeur et 48 en hauteur. Un caractère graphique occupant 8 × 8 pixels, on peut donc définir 10 (80/8) caractères en largeur, et 6 (48/8) caractères en hauteur.

Programme

```
100 REM Programmation de caracteres graphiques m
ultiples
110 :
200 REM Initialisation
210 :
220 CLEAR,,16
230 DEFGR$(0)=&HF0,&HF0,&HF0,&HF0,0,0,0,0
240 DEFGR$(1)=&H0F,&H0F,&H0F,&H0F,0,0,0,0
250 DEFGR$(2)=&HFF,&HFF,&HFF,&HFF,0,0,0,0
260 DEFGR$(3)=0,0,0,0,&HF0,&HF0,&HF0,&HF0
270 DEFGR$(4)=&HF0,&HF0,&HF0,&HF0,&HF0,&HF0,&HF0
,&HF0
280 DEFGR$(5)=&H0F,&H0F,&H0F,&H0F,&HF0,&HF0,&HF0
,&HF0
290 DEFGR$(6)=&HFF,&HFF,&HFF,&HFF,&HF0,&HF0,&HF0
,&HF0
300 DEFGR$(7)=0,0,0,0,&H0F,&H0F,&H0F,&H0F
310 DEFGR$(8)=&HF0,&HF0,&HF0,&HF0,&H0F,&H0F,&H0F
,&H0F
320 DEFGR$(9)=&H0F,&H0F,&H0F,&H0F,&H0F,&H0F,&H0F
,&H0F
330 DEFGR$(10)=&HFF,&HFF,&HFF,&HFF,&H0F,&H0F,&H0
F,&H0F
340 DEFGR$(11)=0,0,0,0,&HFF,&HFF,&HFF,&HFF
350 DEFGR$(12)=&HF0,&HF0,&HF0,&HF0,&HFF,&HFF,&HF
F,&HFF
360 DEFGR$(13)=&H0F,&H0F,&H0F,&H0F,&HFF,&HFF,&HF
F,&HFF
370 DEFGR$(14)=&HFF,&HFF,&HFF,&HFF,&HFF,&HFF,&HF
F,&HFF
375 DEFGR$(15)=0,0,0,0,0,0,0,0
380 :
390 DIM T(48,80)
400 X=1:Y=1 'Position de depart
410 SCREEN 6,0,0
420 :
430 REM ***************************************
500 REM Gestion du curseur
510 :
```

```
511 CLS
520 LOCATE INT((X-1)/2),INT((Y-1)/2)
530 A$=INKEY$:IF A$="" THEN 530
540 A=ASC(A$)
550 IF A$="P" THEN T(Y,X)=1:CP=1:GOSUB 1000:PRIN
T GR$(CA) 'Carre Plein
560 IF A$="V" THEN T(Y,X)=0:CP=0:GOSUB 1000:PRIN
T GR$(CA) 'Carre Vide
570 IF A=9  AND X<80 THEN X=X+1
580 IF A=8  AND X>1  THEN X=X-1
590 IF A=10 AND Y<48 THEN Y=Y+1
600 IF A=11 AND Y>1  THEN Y=Y-1
610 IF A<>8 AND A<>9 AND A<>10 AND A<>11 AND A<>
80 AND A<>86 AND A<>13 THEN PLAY"DO"
620 IF A<>13 THEN 520 'Boucle de saisie
630 REM ***************************************
700 REM Affichage des resultats
710 :
720 CLS
730 PRINT"Sortie des resultats :"
740 PRINT"1)Sur Ecran,"
750 PRINT"2)Sur Imprimante,"
760 PRINT:INPUT "Votre choix :";R
770 IF R<>1 AND R<>2 THEN 700 'Reponse refusee
780 IF R=1 THEN OPEN "O",#1,"SCRN:"
785 IF R=2 THEN OPEN "O",#1,"LRPT:"
790 :
800 FOR I=1 TO 6
810   FOR J=1 TO 10
820     PRINT#1:PRINT#1,"Lig."I"Col."J":";
830     FOR K=1 TO 8
840       A=0
850       FOR L=1 TO 8
860         A=A+(2^(8-L))*T(K+(I-1)*8,L+(J-1)*8)
870       NEXT L
880       PRINT#1,A;
890     NEXT K
900   NEXT J
910 NEXT I
920 :
930 END
940 REM ***************************************
1000 REM Carre Plein (CP=1) ou Vide(CP=0)
1010 :
1015 S=0 'RAZ Parametre de calcul
1020 IF INT(X/2)<>X/2 AND INT(Y/2)<>Y/2 THEN GOS
UB 1100
1030 IF INT(X/2)= X/2 AND INT(Y/2)<>Y/2 THEN GOS
UB 1200
1035 IF INT(X/2)<>X/2 AND INT(Y/2)= Y/2 THEN GOS
UB 1300
1040 IF INT(X/2)= X/2 AND INT(Y/2)= Y/2 THEN GOS
UB 1400
1050 CA=S-1 'Caractere a afficher
```

```
1055 IF CA<0 THEN CA=15
1060 :
1070 RETURN
1080 REM ***************************************
1100 REM Caractere en haut a gauche
1110 :
1120 IF CP=1 THEN S=S+1
1130 IF T(Y,X+1)=1   THEN S=S+2
1140 IF T(Y+1,X)=1   THEN S=S+4
1150 IF T(Y+1,X+1)=1 THEN S=S+8
1160 :
1170 RETURN
1180 REM ***************************************
1200 REM Caractere en haut a droite
1210 :
1220 IF CP=1 THEN S=S+2
1230 IF T(Y,X-1)=1   THEN S=S+1
1240 IF T(Y+1,X-1)=1 THEN S=S+4
1250 IF T(Y+1,X)=1   THEN S=S+8
1260 :
1270 RETURN
1280 REM ***************************************
1300 REM Caractere en bas a gauche
1310 :
1320 IF CP=1 THEN S=S+4
1330 IF T(Y-1,X)=1   THEN S=S+1
1340 IF T(Y-1,X+1)=1 THEN S=S+2
1350 IF T(Y,X+1)=1   THEN S=S+8
1360 :
1370 RETURN
1380 REM ***************************************
1400 REM Caractere en bas a droite
1410 :
1420 IF CP=1 THEN S=S+8
1430 IF T(Y-1,X-1)=1 THEN S=S+1
1440 IF T(Y-1,X)=1   THEN S=S+2
1450 IF T(Y,X-1)=1   THEN S=S+4
1460 :
1470 RETURN
```

Analyse du programme

Lignes 100 à 410	: Initialisation du programme.
Lignes 500 à 620	: Gestion du curseur sur l'écran.
Lignes 700 à 910	: Affichage des codes DEFGR$.
Lignes 1000 à 1470	: Calcul du caractère GR$ à afficher sur l'écran.

Les trois programmes qui suivent montrent :

- pour le 1er, comment afficher un motif graphique créé par le programme "Définition de caractères graphiques multiples" ;
- pour le 2e, comment animer un tel motif graphique ;
- pour le 3e, comment créer un jeu mettant en œuvre des caractères graphiques.

UTILISATION DES CARACTÈRES GRAPHIQUES AFFICHAGE MONOCHROME D'OBJETS GRAPHIQUES

Si la surface occupée par les objets à afficher n'est pas trop grande, (< 64 × 128 pixels), l'utilisation des caractères graphiques peut s'avérer intéressante.

L'exemple qui suit montre comment afficher le héros d'un jeu de "Heroic Fantasy". Ce personnage est un caractère graphique multiple de 2 caractères de large et de 8 caractères de haut.

Programme

```
10 REM Affichage monochrome d'objets graphiques
20 :
30 CLS
40 CLEAR,,2*8:X=2:Y=8:U=10:V=10
50 GOSUB 1000 'Definition du motif
60 GOSUB 2000 'Affichage du motif
70 END
80 REM **************************************
100 REM Donnees correspondant a l'objet a dessin
er
110 :
120 DATA &HFF,&HFF,&HFC,&HF1,&HE0,&HE0,&HD0,&HC1
130 DATA 0,0,0,0,&H80,&H80,0,0
140 DATA &HC8,&HC7,0,&H78,&HFC,&HBE,&HBF,&HBD
150 DATA 0,0,0,0,0,0,0,0
160 DATA &HBD,&HA9,&H81,&HB9,&H42,&H42,&H42,&H43
170 DATA 0,0,0,0,0,0,0,&HA
180 DATA &H40,&HA0,&HE0,&HFF,&HFF,&HFF,&H80,&H80
190 DATA &HEF,&HFF,&HFF,0,0,0,&H80,&H80
200 DATA &H80,&H80,&H88,&H88,&H88,&H88,&H4A,&HFF
210 DATA &H80,&H80,&H80,&H80,&H80,&H80,&H80,0
220 DATA &H6F,&H6F,&H6F,&H6F,&H37,&H37,&H37,&H37
230 DATA &H80,&H80,&H80,&H80,&H80,&H80,&H80,&H80
240 DATA &H3B,&H3B,&H77,&H77,&H77,&H3B,&H37,&H17
250 DATA &H80,&H80,&HC0,&HC0,&HC0,&H80,&H80,0
260 DATA &H16,&H2F,&H2F,&H2F,0,0,0,0
270 DATA 0,0,&HE0,&HE0,0,0,0,0
280 :
1000 REM **************************************
1001 REM *                                    *
1002 REM *   DEFINITION D'UN MOTIF GRAPHIQUE  *
1003 REM *                                    *
```

```
1004 REM ***************************************
1005 REM *                                     *
1006 REM * Entree : Donnees correspondant au   *
1007 REM *          motif graphique            *
1008 REM *                                     *
1009 REM ***************************************
1010 :
1030 FOR I=1 TO X*Y
1040   FOR J=1 TO 8
1050     READ A(J)
1060   NEXT J
1065   DEFGR$(I-1)=A(1),A(2),A(3),A(4),A(5),A(6)
,A(7),A(8)
1070 NEXT I
1080 :
1090 RETURN
2000 REM ***************************************
2001 REM *                                     *
2002 REM *   AFFICHAGE D'OBJETS GRAPHIQUES     *
2003 REM *                                     *
2004 REM ***************************************
2005 REM *                                     *
2006 REM * Entree : X,Y Dimension de l'objet   *
2007 REM *          U,V Position de l'objet    *
2008 REM *          GR$ Precedemment definis   *
2009 REM * Sortie : Affichage de l'objet       *
2010 REM *                                     *
2011 REM ***************************************
2012 :
2020 FOR I=1 TO Y
2030   FOR J=1 TO X
2040     LOCATE U+J-1,V+I-1
2050     PRINT GR$(J-1+(I-1)*X)
2060   NEXT J
2070 NEXT I
2080 :
2090 RETURN
```

Analyse du programme

Ligne 40	: Définition du nombre de caractères graphiques. X=largeur du héros, Y=hauteur du héros. U,Y = position X et Y du caractère multiple.
Ligne 50	: Définition du caractère graphique.
Ligne 60	: Affichage du caractère graphique.
Lignes 120 à 270	: Données correspondant au héros.
Lignes 1000 à 1090	: Définition du caractère graphique par DEFGR$.
Lignes 2000 à 2090	: Affichage du caractère graphique par GR$ en U,V.

ANIMATION MONOCHROME D'OBJETS GRAPHIQUES

Cette deuxième étape montre comment animer un personnage (en l'occurrence notre héros) en le déplaçant d'un mouvement rectiligne uniforme vers la droite. Cette opération demande la définition de deux objets graphiques :

- Le héros au repos ;
- Le héros en marche, un pied en avant.

La succession de ces deux motifs donnera une impression assez réaliste du déplacement.

Une routine supplémentaire est introduite. Il s'agit de l'effacement d'objets graphiques qui fera disparaître le héros de l'écran.

Programme

```
10 REM Animation monochrome
20 :
30 REM Initialisation
31 :
32 SCREEN 6,0,0:CLS:PRINT SPC(8)"ANIMATION  MONO
CHROME"
33 LOCATE 1,1,0:CLEAR,,32 '32 Symboles graphique
s
34 SY=0 :N=16:GOSUB 1000 'Heros au repos
35 SY=16:N=16:GOSUB 1000 'Heros en marche
36 X=2:Y=8 'Position initiale du heros
37 :
38 REM Deplacement du heros
39 :
40 FOR Z=1 TO 30 STEP 2
41   U=Z:V=10:SY=0:GOSUB 2000 'Affichage
42   GOSUB 3000 'Effacement
43   U=U+1:SY=16:GOSUB 2000 'Affichage
44   GOSUB 3000 'Effacement
45 NEXT Z
46 CLS:END
47 REM ************************************
100 REM Donnees correspondant a l'objet a dessin
er
110 :
120 DATA &HFF,&HFF,&HFC,&HF1,&HE0,&HE0,&HD0,&HC1
130 DATA 0,0,0,0,&H90,&H90,0,0
140 DATA &HC8,&HC7,0,&H78,&HFC,&HBE,&HBF,&HBD
150 DATA 0,0,0,0,0,0,0,0
```

```
160 DATA &HBD,&HA9,&H81,&HB9,&H42,&H42,&H42,&H43
170 DATA 0,0,0,0,0,0,0,&HA
180 DATA &H40,&HA0,&HE0,&HFF,&HFF,&HFF,&H80,&H80
190 DATA &HEF,&HFF,&HFF,0,0,0,&H80,&H80
200 DATA &H80,&H80,&H88,&H88,&H88,&H88,&H4A,&HFF
210 DATA &H80,&H80,&H80,&H80,&H80,&H80,&H80,0
220 DATA &H6F,&H6F,&H6F,&H6F,&H37,&H37,&H37,&H37
230 DATA &H80,&H80,&H80,&H80,&H80,&H80,&H80,&H80
240 DATA &H3B,&H3B,&H77,&H77,&H77,&H3B,&H37,&H17
250 DATA &H80,&H80,&HC0,&HC0,&HC0,&H80,&H80,&H0
260 DATA &H16,&H2F,&H2F,&H2F,0,0,0,0
270 DATA 0,0,&HE0,&HE0,0,0,0,0
280 :
290 REM ************************************
320 DATA &HFF,&HFF,&HFC,&HF1,&HE0,&HE0,&HD0,&HC1
330 DATA 0,0,0,0,&H80,&H80,0,0
340 DATA &HC8,&HC7,0,&H78,&HFC,&HFC,&HFE,&HFD
350 DATA 0,0,0,0,0,0,&HC,&HE
360 DATA &HFD,&HD0,&HCD,&HF1,&H41,&H41,&H41,&H41
370 DATA &H9F,&HCF,&HEE,&HF7,&HFF,&H7E,&H3C,&H18
380 DATA &H41,&H41,&H41,&HFF,&HFF,&HFF,&H80,&H80
390 DATA 0,0,0,0,0,0,&H80,&H80
400 DATA &H80,&H80,&H88,&H88,&H88,&H88,&H4A,&HFF
410 DATA &H80,&H80,&H80,&H80,&H80,&H80,&H80,0
420 DATA &H3D,&H3D,&H3D,&H3D,&H3D,&H7C,&H78,&H70
430 DATA &HC0,&HE0,&HE0,&HE0,&HF0,&H70,&H70,&H70
440 DATA &H70,&H70,&HF8,&HF8,&HF8,&H70,&H70,&H70
450 DATA &H70,&H70,&HFC,&HFC,&HFC,&H38,&H38,&H38
460 DATA &H70,&HF8,&HFE,&HFE,0,0,0,0
470 DATA &H38,&H7C,&H7F,&H7F,0,0,0,0
1000 REM *************************************
1001 REM *                                   *
1002 REM *   DEFINITION D'UN MOTIF GRAPHIQUE *
1003 REM *                                   *
1004 REM *************************************
1005 REM *                                   *
1006 REM * Entree : Donnees correspondant au *
1007 REM *          motif graphique          *
1008 REM *          SY=Symbole de debut de   *
1009 REM *          definition               *
1010 REM *          N =Nombre de caracteres  *
1011 REM *          redefinis                *
1012 REM *                                   *
1013 REM *************************************
1014 :
1030 FOR I=1 TO N
1040   FOR J=1 TO 8
1050     READ A(J)
1060   NEXT J
```

```
1065   DEFGR$(SY+I-1)=A(1),A(2),A(3),A(4),A(5),A
(6),A(7),A(8)
1070 NEXT I
1080 :
1090 RETURN
2000 REM ***************************************
2001 REM *                                     *
2002 REM *   AFFICHAGE D'OBJETS GRAPHIQUES     *
2003 REM *                                     *
2004 REM ***************************************
2005 REM *                                     *
2006 REM * Entree : X,Y Dimension de l'objet   *
2007 REM *          U,V Position de l'objet    *
2008 REM *          GR$ precedemment definis   *
2009 REM * Sortie : Affichage de l'objet       *
2010 REM *                                     *
2011 REM ***************************************
2012 :
2020 FOR I=1 TO Y
2030   FOR J=1 TO X
2040     LOCATE U+J-1,V+I-1
2050     PRINT GR$(SY+J-1+(I-1)*X)
2060   NEXT J
2070 NEXT I
2080 :
2090 RETURN
3000 REM ***************************************
3001 REM *                                     *
3002 REM *    EFFACEMENT D'OBJETS GRAPHIQUES   *
3003 REM *                                     *
3004 REM ***************************************
3005 REM *                                     *
3006 REM * Entree : X,Y=Dimension de l'objet   *
3007 REM *          U,V=Position de l'objet    *
3008 REM * Sortie : Effacement de l'objet      *
3009 REM *                                     *
3010 REM ***************************************
3011 :
3020 FOR I=1 TO Y
3030   LOCATE U,V+I-1
3040   PRINT SPC(X)
3050 NEXT I
3060 :
3070 RETURN
```

Analyse du programme

Ligne 33	: Définition de 8×2×2 symboles graphiques, effacement du curseur.
Ligne 34	: Définition du héros au repos par DEFGR$.
Ligne 35	: Définition du héros en marche par DEFGR$.
Ligne 36	: Héros dans sa position initiale.
Lignes 40 à 45	: Déplacement du héros.
Ligne 41	: Affichage du héros au repos.
Ligne 42	: Effacement du héros au repos.
Ligne 43	: Affichage du héros en marche.
Ligne 44	: Effacement du héros en marche.
Lignes 120 à 270	: Héros au repos.
Lignes 320 à 470	: Héros en marche.
Lignes 1000 à 1090	: Définition d'un motif graphique par DEFGR$.
Lignes 2000 à 2090	: Affichage d'un motif graphique par GR$.
Lignes 3000 à 3070	: Effacement d'un motif graphique.

JEU D'ANIMATION MONOCHROME

La dernière étape montre comment créer un jeu mettant en œuvre des objets graphiques, et dans lequel le joueur intervient en tapant des ordres en "temps réel" (sans attente visible) pour orienter les objets à déplacer.

Notre – désormais célèbre – héros est repris ici ; Il peut se déplacer vers la droite et vers la gauche avec deux positions différentes dans chaque direction (il faut donc définir 4 objets graphiques).

Le but de ce jeu est d'attraper des pièces d'or qui tombent du ciel en se plaçant juste au-dessous. Un bip sonore indique chaque récupération correctement effectuée.

Le joueur peut déplacer le héros vers la droite (touche-flèche droite), vers la gauche (touche-flèche gauche) ou arrêter la partie (touche ENTER).

Programme

```
10 REM Jeu d'animation monochrome
20 :
30 REM Initialisation
31 :
32 SCREEN 6,0,0:CLS
33 LOCATE 1,1,0:CLEAR,,65 '66 Symboles graphique
s
```

```
34 SY=0 :N=16:GOSUB 1000 'Heros au repos droit
35 SY=16:N=16:GOSUB 1000 'Heros en marche droit
36 SY=32:N=16:GOSUB 1000 'Heros au repos gauche
37 SY=48:N=16:GOSUB 1000 'Heros en marche gauche
38 SY=64:N=1 :GOSUB 1000 'Piece carree
39 X=2:Y=8:V=15:U=1:XP=1:PO=0 'Initialisation
40 :
41 REM Deplacement du heros
42 :
43 A$=INKEY$ 'Lecture clavier
44 IF A$<>"" THEN A=ASC(A$)
45 IF A=9 THEN GOSUB 4000 'Deplacement vers la D
roite
46 IF A=8 THEN GOSUB 5000 'Deplacement vers la G
auche
47 GOSUB 6000 'Deplacement des Pieces Carrees
48 IF A<>13 THEN 43 'Boucle de jeu
49 CLS:END
50 REM *************************************
100 REM Donnees correspondant a l'objet a dessin
er
110 :
120 DATA &HFF,&HFF,&HFC,&HF1,&HE0,&HE0,&HD0,&HC1
130 DATA 0,0,0,0,&H80,&H80,0,0
140 DATA &HC8,&HC7,0,&H78,&HFC,&HBE,&HBF,&HBD
150 DATA 0,0,0,0,0,0,0,0
160 DATA &HBD,&HA9,&H81,&HB9,&H42,&H42,&H42,&H43
170 DATA 0,0,0,0,0,0,0,&HA
180 DATA &H40,&HA0,&HE0,&HFF,&HFF,&HFF,&H80,&H80
190 DATA &HEF,&HFF,&HFF,0,0,0,&H80,&H80
200 DATA &H80,&H80,&H88,&H88,&H88,&H88,&H4A,&HFF
210 DATA &H80,&H80,&H80,&H80,&H80,&H80,&H80,0
220 DATA &H6F,&H6F,&H6F,&H6F,&H37,&H37,&H37,&H37
230 DATA &H80,&H80,&H80,&H80,&H80,&H80,&H80,&H80
240 DATA &H3B,&H3B,&H77,&H77,&H77,&H3B,&H37,&H17
250 DATA &H80,&H80,&HC0,&HC0,&HC0,&H80,&H80,&H0
260 DATA &H16,&H2F,&H2F,&H2F,0,0,0,0
270 DATA 0,0,&HE0,&HE0,0,0,0,0
280 :
290 REM *************************************
320 DATA &HFF,&HFF,&HFC,&HF1,&HE0,&HE0,&HD0,&HC1
330 DATA 0,0,0,0,&H80,&H80,0,0
340 DATA &HC8,&HC7,0,&H78,&HFC,&HFC,&HFE,&HFD
350 DATA 0,0,0,0,0,0,&HC,&HE
360 DATA &HFD,&HD0,&HCD,&HF1,&H41,&H41,&H41,&H41
370 DATA &H9F,&HCF,&HEE,&HF7,&HFF,&H7E,&H3C,&H18
380 DATA &H41,&H41,&H41,&HFF,&HFF,&HFF,&H80,&H80
390 DATA 0,0,0,0,0,0,&H80,&H80
400 DATA &H80,&H80,&H88,&H88,&H88,&H88,&H4A,&HFF
410 DATA &H80,&H80,&H80,&H80,&H80,&H80,&H80,0
420 DATA &H3D,&H3D,&H3D,&H3D,&H3D,&H7C,&H78,&H70
```

```
430 DATA &HC0,&HE0,&HE0,&HE0,&HF0,&H70,&H70,&H70
440 DATA &H70,&H70,&HF8,&HF8,&HF8,&H70,&H70,&H70
450 DATA &H70,&H70,&HFC,&HFC,&HFC,&H38,&H38,&H38
460 DATA &H70,&HF8,&HFE,&HFE,0,0,0,0
470 DATA &H38,&H7C,&H7F,&H7F,0,0,0,0
480 REM ****************************************
520 DATA 0,0,0,0,1,1,0,0
530 DATA &HFF,&HFF,&H3F,&H5F,7,7,&HB,&H83
540 DATA 0,0,0,0,0,0,0,0
550 DATA &H13,&HE3,0,&H1E,&H3F,&H7D,&HFD,&HBD
560 DATA 0,0,0,0,0,0,0,&H50
580 DATA &HBD,&H95,&H81,&H9D,&H42,&H42,&H42,&HC2
590 DATA &HF7,&HFF,&HFF,0,0,0,1,1
600 DATA 2,6,6,&HFF,&HFF,&HFF,1,1
610 DATA 1,1,1,1,1,1,1,0
620 DATA 1,1,&H11,&H11,&H11,&H11,&H52,&HFE
630 DATA 1,1,1,1,1,1,1,1
640 DATA &HF6,&HF6,&HF6,&HF6,&HEC,&HEC,&HEC,&HEC
650 DATA 1,1,3,3,3,1,1,0
660 DATA &HDC,&HDC,&HEE,&HEE,&HEE,&HDC,&HEC,&HE8
670 DATA 0,0,7,7,0,0,0,0
680 DATA &H68,&HF4,&HF4,&HF4,0,0,0,0
690 REM ****************************************
720 DATA 0,0,0,0,1,1,0,0
730 DATA &HFF,&HFF,&H3F,&H5F,7,7,&HB,&H83
740 DATA 0,0,0,0,0,0,&H30,&H70
750 DATA &H13,&HE3,0,&H1E,&H3F,&H3F,&H7F,&HBF
760 DATA &HF9,&HF3,&H77,&HEF,&HFF,&H7E,&H3C,&H18
780 DATA &HBF,&HBB,&HB7,&H8F,&H11,&H81,&H81,&H81
790 DATA 0,0,0,0,0,0,1,1
800 DATA &H81,&H81,&H81,&HFF,&HFF,&HFF,1,1
810 DATA 1,1,1,1,1,1,1,0
820 DATA 1,1,&H11,&H11,&H11,&H11,&H52,&HFE
830 DATA 3,7,7,7,&HF,&HE,&HE,&HE
840 DATA &HBC,&HBC,&HBC,&HBC,&HBC,&H3E,&H1E,&HE
850 DATA &HE,&HE,&H3F,&H3F,&H3F,&H1C,&H1C,&H1C
860 DATA &HE,&HE,&H1F,&H1F,&H1F,&HE,&HE,&HE
870 DATA &H1C,&H3E,&HFE,&HFE,0,0,0,0
880 DATA &HE,&H1F,&H7F,&H7F,0,0,0,0
890 REM ****************************************
900 DATA &HFF,&H81,&H81,&H81,&H81,&H81,&H81,&HFF
1000 REM ***************************************
1001 REM *                                     *
1002 REM *   DEFINITION D'UN MOTIF GRAPHIQUE   *
1003 REM *                                     *
1004 REM ***************************************
1005 REM *                                     *
1006 REM * Entree : Donnees correspondant au   *
1007 REM *          motif graphique            *
1008 REM *          SY=Symbole de debut de     *
1009 REM *          definition                 *
```

```
1010 REM *           N =Nombre de caracteres    *
1011 REM *           redefinis                  *
1012 REM *                                      *
1013 REM ****************************************
1014 :
1030 FOR I=1 TO N
1040   FOR J=1 TO 8
1050     READ A(J)
1060   NEXT J
1065   DEFGR$(SY+I-1)=A(1),A(2),A(3),A(4),A(5),A
(6),A(7),A(8)
1070 NEXT I
1080 :
1090 RETURN
2000 REM ****************************************
2001 REM *                                      *
2002 REM *   AFFICHAGE D'OBJETS GRAPHIQUES      *
2003 REM *                                      *
2004 REM ****************************************
2005 REM *                                      *
2006 REM * Entree : X,Y Dimension de l'objet    *
2007 REM *          U,V Position de l'objet     *
2008 REM *          GR$ precedemment definis    *
2009 REM * Sortie : Affichage de l'objet        *
2010 REM *                                      *
2011 REM ****************************************
2012 :
2020 FOR I=1 TO Y
2030   FOR J=1 TO X
2040     LOCATE U+J-1,V+I-1
2050     PRINT GR$(SY+J-1+(I-1)*X)
2060   NEXT J
2070 NEXT I
2080 :
2090 RETURN
3000 REM ****************************************
3001 REM *                                      *
3002 REM *    EFFACEMENT D'OBJETS GRAPHIQUES    *
3003 REM *                                      *
3004 REM ****************************************
3005 REM *                                      *
3006 REM * Entree : X,Y=Dimension de l'objet    *
3007 REM *          U,V=Position de l'objet     *
3008 REM * Sortie : Effacement de l'objet       *
3009 REM *                                      *
3010 REM ****************************************
3011 :
3020 FOR I=1 TO Y
3030   LOCATE U,V+I-1
3040   PRINT SPC(X)
3050 NEXT I
```

```
3060 :
3070 RETURN
4000 REM ************************************
4001 REM *                                  *
4002 REM * DEPLACEMENT DU HEROS VERS LA DROITE *
4003 REM *                                  *
4004 REM ************************************
4005 :
4010 IF U>35 THEN RETURN 'Deplacement refuse
4020 :
4030 IF P=1 THEN P=0 ELSE P=1 'Position repos ou
 marche
4040 GOSUB 3000 'Effacement position precedente
4050 IF P=1 THEN SY=0 ELSE SY=16
4060 U=U+2:GOSUB 2000 'Deplacement du heros
4070 :
4080 RETURN
5000 REM ************************************
5001 REM *                                  *
5002 REM * DEPLACEMENT DU HEROS VERS LA GAUCHE *
5003 REM *                                  *
5004 REM ************************************
5005 :
5010 IF U<3 THEN RETURN 'Deplacement refuse
5020 :
5030 IF P=1 THEN P=0 ELSE P=1 'Position repos ou
 marche
5040 GOSUB 3000 'Effacement position precedente
5050 IF P=1 THEN SY=32 ELSE SY=48
5060 U=U-2:GOSUB 2000 'Deplacement du heros
5070 :
5080 RETURN
6000 REM ************************************
6001 REM *                                  *
6002 REM * DEPLACEMENT DES PIECES CARREES   *
6003 REM *                                  *
6004 REM ************************************
6005 :
6010 IF A=32 AND RND<0.8 THEN RETURN
6020 IF PO>0 THEN LOCATE XP,PO:PRINT" "
6030 IF PO>10 AND ABS(XP-U)<2 THEN PLAY"DO" 'Un
point
6040 IF PO>10 THEN PO=0
6050 IF PO=0 THEN XP=INT(RND*30)+5
6060 PO=PO+2
6070 LOCATE XP,PO:PRINT GR$(64)
6080 :
6090 RETURN
```

Analyse du programme

Ligne 33	: Effacement curseur et définition de 8×2×4 symboles graphiques.
Ligne 34	: Définition du héros au repos tourné vers la droite.
Ligne 35	: Définition du héros en marche tourné vers la droite.
Ligne 36	: Définition du héros au repos tourné vers la gauche.
Ligne 37	: Définition du héros en marche tourné vers la gauche.
Ligne 38	: Définition de la pièce d'or.
Ligne 39	: Initialisation.
Ligne 43	: Lecture du clavier.
Ligne 45	: Déplacement vers la droite demandé.
Ligne 46	: Déplacement vers la gauche demandé.
Ligne 47	: Déplacement de la pièce d'or.
Ligne 48	: Boucle de jeu.
Lignes 100 à 900	: Données correspondant au héros et aux P.O.
Lignes 1000 à 1090	: Définition d'un motif graphique par DEFGR$.
Lignes 2000 à 2090	: Affichage d'un motif graphique par GR$.
Lignes 3000 à 3070	: Effacement d'un motif graphique.
Lignes 4000 à 4080	: Déplacement du héros vers la droite.
Lignes 5000 à 5080	: Déplacement du héros vers la gauche.
Lignes 6000 à 6090	: Déplacement des P.O. (pièces d'or).

CONSTITUTION D'IMAGES ÉCRAN AVEC LE CRAYON OPTIQUE

Une manière simple pour constituer vos décors ou personnages fixes consiste à se servir du crayon optique. Cet article se révèle toutefois un peu imprécis et demande beaucoup de soins pour donner un dessin au graphisme acceptable.

Vous sélectionnez la couleur d'affichage avec les touches A à O pour MO5 et avec les touches A à H pour TO7/70.

Programme

```
10000 REM Crayonnage couleur
10001 :
10010 SCREEN 6,3,3:CLS:LOCATE 1,1,0
10020 C=4 'Init couleur trace
10025 :
```

```
10030 INPUTPEN X,Y
10040 IF X=-1 OR Y=-1 THEN 10030
10050 PSET(X,Y),C
10060 A$=INKEY$
10070 IF A$="" THEN 10030
10080 IF A$>="A" AND A$<="O" THEN C=ASC(A$)-64:B
EEP
10090 IF ASC(A$)<>13 THEN 10030
10100 END
```

Analyse du programme

Ligne 10010	: Disparition du curseur.
Ligne 10020	: Initialisation de la couleur du tracé.
Ligne 10030	: Lecture du crayon optique.
Ligne 10040	: Lecture erronée.
Ligne 10050	: Affichage d'un point élémentaire sur l'écran.
Ligne 10060	: Lecture d'une touche du clavier.
Ligne 10070	: Boucle de programme.
Ligne 10080	: Changement de couleur.
Ligne 10090	: Fin de programme ou boucle.

Pour terminer ce chapitre sur l'animation graphique, voici quelques programmes écrits en assembleur et incorporés dans des programmes BASIC sous la forme de DATA. Ces programmes vous permettront :

- de sauvegarder et restituer la page mémoire d'écran ;
- de rechercher un octet ou une suite de deux octets en mémoire.

SAUVEGARDE ET RESTITUTION DE CONTEXTE

Sauvegarde monochrome

La page écran est sauvegardée à partir de l'adresse mémoire &H4000.

Programme Assembleur

```
   8E 40 00        LDX  #$4000 * Adresse Debut Sauvegarde
10 8E 00 00        LDY  #$0000 * Adresse Debut Ecran
      A6 A0 BOUCLE LDA  ,Y+    * Lecture Ecran
      A7 80        STA  ,X+    * Ecriture Sauvegarde
10 8C 1F 40        CMPY #$1F40 * Fin d'ecriture ?
      26 F6        BNE  BOUCLE * Non
         39        RTS         * Oui
```

Programme

```
100 REM Sauvegarde de contexte monochrome
110 :
120 FOR I=0 TO 17
130   READ A:POKE &H1F40+I,A
140 NEXT I
150 :
160 DATA &H8E,&H40,0,&H10,&H8E,0,0,&HA6,&HA0,&HA
7,&H80
170 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
180 :
190 POKE &HA7C0,PEEK(&HA7C0) OR 1 'Selection mem
 graphique
200 EXEC &H1F40 'Sauvegarde
```

Analyse du programme BASIC

Lignes 120 à 140	: Mise en mémoire de la routine ASM.
Lignes 160 à 170	: DATA de la routine ASM.
Ligne 190	: Sélection de la mémoire motif.
Ligne 200	: Sauvegarde.

Restitution monochrome

L'opération inverse de la précédente consiste à restituer l'écran précédemment sauvegardé. Le programme ASM est sensiblement ressemblant au précédent.

Programme Assembleur

```
   8E 40 00        LDX  #$4000 * Adresse Debut Sauvegarde
10 8E 00 00        LDY  #$0000 * Adresse Debut Ecran
      A6 80 BOUCLE LDA  ,X+    * Lecture Sauvegarde
      A7 A0        STA  ,Y+    * Ecriture Ecran
10 8C 1F 40        CMPY #$1F40 * Fin d'ecriture ?
      26 F6        BNE  BOUCLE * Non
         39        RTS         * Oui
```

Programme

```
100 REM Lecture de contexte monochrome
110 :
120 FOR I=0 TO 17
130   READ A:POKE &H1F40+I,A
140 NEXT I
150 :
```

```
160 DATA &H8E,&H40,0,&H10,&H8E,0,0,&HA6,&H80,&HA
7,&HA0
170 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
180 :
190 POKE &HA7C0,PEEK(&HA7C0) OR 1 'Selection mem
 graphique
200 EXEC &H1F40 'Lecture
```

Analyse du programme BASIC

Lignes 120 à 140	: Mise en mémoire de la routine ASM.
Lignes 160 à 170	: DATA de la routine ASM.
Ligne 190	: Sélection de la mémoire Motif.
Ligne 200	: Restitution de l'écran sauvegardé.

Sauvegarde multichrome

La mémoire d'écran couleur occupant la même position que la mémoire d'écran motif, la mémoire d'écran couleur étant accédée par

POKE &HA7C0,PEEK(&HA7C0) AND 254

enfin, la mémoire d'écran motif étant accédée par

POKE &HA7C0,PEEK(&HA7C0) OR 1

il est très facile de faire une sauvegarde d'écran couleur en reprenant le procédé précédent et en forçant successivement à 0 et à 1 le bit 0 de la mémoire &HA7C0.

Programme ASM

Sauvegarde Motif : même programme que pour la sauvegarde monochrome.

Sauvegarde Couleur :

```
   8E 60 00          LDX  #$6000 * Adresse Debut Sauvegarde
10 8E 00 00          LDY  #$0000 * Adresse Debut Ecran
      A6 80 BOUCLE LDA  ,X+     * Lecture Sauvegarde
      A7 A0          STA  ,X+     * Ecriture Ecran
10 8C 1F 40          CMPY #$1F40 * Fin d'ecriture ?
      26 F6          BNE  BOUCLE * Non
         39          RTS          * Oui
```

Programme

```
100 REM Sauvegarde de contexte multichrome
110 :
115 POKE &HA7C0,PEEK(&HA7C0) OR 1 'Selection mot
if
120 FOR I=0 TO 17
130   READ A:POKE &H1F40+I,A
140 NEXT I
150 :
160 DATA &H8E,&H40,0,&H10,&H8E,0,0,&HA6,&HA0,&HA
7,&H80
170 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
180 :
200 EXEC &H1F40 'Sauvegarde Motif
210 :
212 POKE &HA7C0,PEEK(&HA7C0) AND 254 'Selection
couleur
220 FOR I=0 TO 17
230   READ A:POKE &H1F40+I,A
240 NEXT I
250 :
260 DATA &H8E,&H60,0,&H10,&H8E,0,0,&HA6,&HA0,&HA
7,&H80
270 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
280 :
300 EXEC &H1F40 'Sauvegarde Couleur
```

Analyse du programme BASIC

Ligne 115	: Sélection de la mémoire d'écran motif.
Lignes 120 à 140	: Mise en mémoire du programme ASM restitution du motif.
Lignes 160 à 170	: DATA du programme ASM sauvegarde de motif.
Ligne 200	: Sauvegarde du motif.
Lignes 220 à 240	: Mise en mémoire du programme ASM de sauvegarde de couleur.
Lignes 260 à 270	: DATA du programme ASM sauvegarde de couleur.
Ligne 300	: Sauvegarde couleur.

Restitution multichrome

L'opération inverse de la précédente permet de restituer l'écran couleur précédemment sauvegardé.

Programme ASM

Restitution Motif : même programme que celui du programme restitution Monochrome.

Restitution Couleur :

```
   8E 60 00         LDX   #$6000 * Adresse Debut Sauvegarde
10 8E 00 00         LDY   #$0000 * Adresse Debut Ecran
      A6 A0  BOUCLE LDA   ,Y+    * Lecture Ecran
      A7 80         STA   ,X+    * Ecriture Sauvegarde
10 8C 1F 40         CMPY  #$1F40 * Fin d'ecriture ?
      26 F6         BNE   BOUCLE * Non
         39         RTS          * Oui
```

Programme

```
100 REM Lecture de contexte multichrome
110 :
115 POKE &HA7C0,PEEK(&HA7C0) OR 1 'Selection mot
if
120 FOR I=0 TO 17
130   READ A:POKE &H1F40+I,A
140 NEXT I
150 :
160 DATA &H8E,&H40,0,&H10,&H8E,0,0,&HA6,&H80,&HA
7,&HA0
170 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
180 :
200 EXEC &H1F40 'Lecture Motif
210 :
212 POKE &HA7C0,PEEK(&HA7C0) AND 254 'Selection
couleur
220 FOR I=0 TO 17
230   READ A:POKE &H1F40+I,A
240 NEXT I
250 :
260 DATA &H8E,&H60,0,&H10,&H8E,0,0,&HA6,&H80,&HA
7,&HA0
270 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
280 :
300 EXEC &H1F40 'Lecture Couleur
```

Analyse du programme BASIC

Ligne 115 : Sélection de la mémoire d'écran motif.
Lignes 120 à 140 : Mise en mémoire du programme ASM de restitution motif.
Lignes 160 à 170 : DATA du programme ASM restitution motif.
Ligne 200 : Lecture motif.
Ligne 212 : Sélection de la mémoire d'écran couleur.
Lignes 220 à 240 : Mise en mémoire du programme ASM restitution couleur.
Lignes 260 à 270 : DATA du programme ASM restitution couleur.
Ligne 300 : Lecture couleur.

Le programme suivant fait une démonstration de sauvegarde puis de restitution d'écran couleur.

Programme

```
10 REM Demonstration de sauvegarde
11 :
12 CLS
13 FOR I=1 TO 15
14   COLOR I
15   PRINT"Demonstration de sauvegarde de CTXT"
16 NEXT I
17 GOSUB 100 'Sauvegarde
18 :
19 CLS : LOCATE 0,0,0
20 PRINT"Appuyez sur une touche pour reafficher"
21 PRINT"l'ecran precedent."
22 A$=INPUT$(1)
23 GOSUB 500
24 LOCATE 0,15
25 :
26 END
27 REM ********************************
100 REM Sauvegarde de contexte multichrome
110 :
115 POKE &HA7C0,PEEK(&HA7C0) OR 1 'Selection mot
if
120 FOR I=0 TO 17
130   READ A:POKE &H1F40+I,A
140 NEXT I
150 :
160 DATA &H8E,&H40,0,&H10,&H8E,0,0,&HA6,&HA0,&HA
7,&H80
170 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
180 :
200 EXEC &H1F40 'Sauvegarde Motif
210 :
212 POKE &HA7C0,PEEK(&HA7C0) AND 254 'Selection
couleur
220 FOR I=0 TO 17
230   READ A:POKE &H1F40+I,A
240 NEXT I
250 :
```

```
260 DATA &H8E,&H60,0,&H10,&H8E,0,0,&HA6,&HA0,&HA
7,&H80
270 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
280 :
300 EXEC &H1F40 'Sauvegarde Couleur
310 :
320 RETURN
330 REM ********************************
500 REM Lecture de contexte multichrome
510 :
515 POKE &HA7C0,PEEK(&HA7C0) OR 1 'Selection mot
if
520 FOR I=0 TO 17
530   READ A:POKE &H1F40+I,A
540 NEXT I
550 :
560 DATA &H8E,&H40,0,&H10,&H8E,0,0,&HA6,&H80,&HA
7,&HA0
570 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
580 :
600 EXEC &H1F40 'Lecture Motif
610 :
612 POKE &HA7C0,PEEK(&HA7C0) AND 254 'Selection
couleur
620 FOR I=0 TO 17
630   READ A:POKE &H1F40+I,A
640 NEXT I
650 :
660 DATA &H8E,&H60,0,&H10,&H8E,0,0,&HA6,&H80,&HA
7,&HA0
670 DATA &H10,&H8C,&H1F,&H40,&H26,&HF6,&H39
680 :
700 EXEC &H1F40 'Lecture Couleur
710 :
720 RETURN
```

Analyse du programme

Ligne 17	: Sauvegarde écran couleur.
Ligne 23	: Lecture écran couleur.
Lignes 100 à 320	: Sauvegarde d'écran couleur : mise en mémoire puis exécution routine sauvegarde motif ; mise en mémoire puis exécution routine sauvegarde couleur.
Lignes 500 à 720	: Restitution d'écran couleur : mise en mémoire puis exécution routine lecture motif ; mise en mémoire puis exécution routine lecture couleur.

Pour terminer ce chapitre, voici deux programmes qui vous permettront de trouver 1 ou 2 octets en mémoire (d'écran par exemple). Ces deux programmes sont écrits en assembleur et incorporés à une procédure BASIC sous la forme de DATA.

RECHERCHE D'UN OCTET EN MÉMOIRE

L'octet à rechercher est mis dans la variable O. L'adresse de début de recherche dans la variable AD. La procédure de recherche est appelée par GOSUB 10000. La 1re adresse contenant l'octet O à partir de l'adresse AD est donnée dans A en sortie du programme.

Programme Assembleur

```
BE 1F 42          LDX  $1F42 * Rech a Partir de 1F42
   A6 80 BOUCLE LDA  ,X+    * Lecture memoire
B1 1F 40          CMPA $1F40 * Donnee recherchee ?
   26 F9          BNE  BOUCLE * Non
BF 1F 44          STX  $1F44  * @ Donnee trouvee
      39          RTS
```

Exemple

```
'
10 AD=&H2000 'Adresse de depart
20 O =11      'Octet recherche
30 GOSUB 10000 ' Recherche
40 PRINT A 'Adresse recherchee
50 END
```

Programme

```
10000 REM ************************************
10001 REM *                                  *
10002 REM *  RECHERCHE D'UN OCTET EN MEMOIRE  *
10003 REM *                                  *
10004 REM ************************************
10005 REM *                                  *
10006 REM * Entree : AD=Adresse de depart    *
10007 REM *          O =Octet recherche      *
10008 REM * Sortie : A =1ere Adresse contenant A*
10009 REM *                                  *
10010 REM ************************************
10011 :
10020 MSB=INT(AD/256):LSB=AD-MSB*256 'Calcul Poid
s faible et fort Adresse
10030 POKE &H1F42,MSB:POKE&H1F43,LSB 'Memorisatio
n Adresse
10040 POKE &H1F40,O 'Memorisation Octet recherche
10045 :
```

```
10050 FOR I=0 TO 13
10060   READ A
10070   POKE &H1F50+I,A
10080 NEXT I
10100 DATA &HBE,&H1F,&H42,&HA6,&H80,&HB1,&H1F,&H4
0,&H26,&HF9,&HBF,&H1F,&H44,&H39
10105 :
10110 EXEC&H1F50 'Recherche
10120 A=PEEK(&H1F44)*256+PEEK(&H1F45)-1
10130 :
10140 RETURN
```

Analyse du programme BASIC

Ligne 10020	: Calcul poids fort et faible de l'adresse de début de recherche.
Ligne 10030	: Mémorisation de cette adresse.
Ligne 10040	: Mémorisation de l'octet recherché.
Lignes 10050 à 10100	: Mémorisation de la routine ASM.
Ligne 10110	: Recherche.
Ligne 10120	: Mémorisation de la 1re adresse contenant O.

RECHERCHE DE DEUX OCTETS CONSÉCUTIFS EN MÉMOIRE

Les deux octets à rechercher sont mis dans les variables O1 et O2. L'adresse de début de recherche dans AD. La procédure est appelée par GOSUB 10000. La 1re adresse contenant l'octet O à partir de l'adresse AD est donnée dans A en sortie de programme.

Programme Assembleur

```
BE 1F 42        LDX  $1F42 * Rech. a partir de 1F42
   A6 80 BOUC1  LDA  ,X+
B1 1F 40 BOUC2  CMPA $1F40 * 1e data rech=data lue?
   26 F9        BNE  BOUC1 * Non
   A6 80        LDA  ,X+   * Oui
B1 1F 41        CMP  $1F41 * 2e data rech=data lue?
   26 F4        BNE  BOUC2 * Non
BF 1F 44        STX  $1F44 * Sauv @ memoire
      39        RTS
```

Exemple

```
10 CLS:POKE &HA7C0,PEEK(&HA7C0) OR 1 'Selection m
otif graphique
20 POKE 7036,23:POKE 7037,67 'Ecriture dans Mem.
ecran
30 AD=0  'Adresse de debut de recherche
40 O1=23 '1er  Octet recherche
50 O2=67 '2eme Octet recherche
60 GOSUB 10000 ' Recherche
70 IF A>=8000 THEN PRINT"Motif absent" ELSE PRINT
"Motif en "A
80 END
```

Programme

```
10000 REM ***************************************
10001 REM *                                     *
10002 REM *  RECHERCHE DE 2 OCTETS EN MEMOIRE   *
10003 REM *                                     *
10004 REM ***************************************
10005 REM *                                     *
10006 REM * Entree : AD=Adresse de depart       *
10007 REM *          O1=1er  octet recherche    *
10008 REM *          O2=2eme octet recherche    *
10009 REM * Sortie : A =1ere Adresse contenant A*
10010 REM *                                     *
10011 REM ***************************************
10012 :
10020 MSB=INT(AD/256):LSB=AD-MSB*256 'Calcul poid
s faible et fort Adresse
10030 POKE &H1F42,MSB:POKE&H1F43,LSB
10040 POKE &H1F40,O1:POKE&H1F41,O2
10045 :
10050 FOR I=0 TO 20
10060   READ A
10070   POKE &H1F50+I,A
10080 NEXT I
10100 DATA &HBE,&H1F,&H42,&HA6,&H80,&HB1,&H1F,&H4
0,&H26,&HF9,&HA6,&H80
10101 DATA &HB1,&H1F,&H41,&H26,&HF4,&HBF,&H1F,&H4
4,&H39
10105 :
10110 EXEC&H1F50 'Recherche
10120 A=PEEK(&H1F44)*256+PEEK(&H1F45)-2
10130 :
10140 RETURN
```

Analyse du programme BASIC

Ligne 10020	: Calcul des poids fort et faible de l'adresse de début de recherche.
Ligne 10030	: Mémorisation de cette adresse.
Ligne 10040	: Mémorisation des octets recherchés.
Lignes 10050 à 10101	: Mémorisation de la routine ASM.
Ligne 10110	: Recherche.
Ligne 10120	: Mémorisation de la 1[re] adresse contenant O1,O2.

Compatibilité des MO5 et TO7/70

Les programmes présentés dans ce manuel ont été écrits sur un MO5. Si vous possédez un TO7/70, certaines modifications doivent être apportées car la compatibilité entre MO5 et TO7/70 n'est pas complète.

En ce qui concerne les programmes décrits dans ce livre, peu de modifications sont à effectuer. Ces modifications concernent :

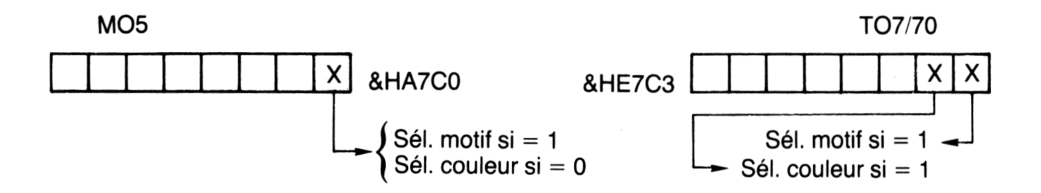
1. - l'octet permettant de sélectionner mémoire d'écran graphique ou mémoire d'écran couleur ;
2. - les adresses de mémoire d'écran ;
3. - l'utilisation des couleurs ;
4. - l'implantation des programmes Assembleur en mémoire.

Les programmes dont la liste suit sont à modifier :

- Copie d'écran monochrome, page 63.
- Copie d'écran multichrome, page 64.
- Couleur d'un caractère de l'écran, page 81.
- Palette de couleurs, page 126.
- TAG, page 127.
- Repère paramétrable, page 129.
- Crayonnage couleur, page 153.
- Sauvegarde de contexte monochrome et multichrome, page 154.
- Lecture de contexte monochrome et multichrome, page 156.
- Recherche d'octet(s) en mémoire, pages 160 à 162.

Pour les curieux, voici les explications détaillées des incompatibilités entre MO5 et TO7/70. (Les incompatibilités décrites restent dans le cadre de ce manuel. Elles ne sont donc pas exhaustives).

1° *Octet de sélection mémoire d'écran graphique/mémoire d'écran couleur*



ce qui se traduit par l'ordre BASIC :

"POKE &HE7C3,PEEK(&HE7C3) OR 1"

pour sélectionner la mémoire d'écran graphique et

"POKE &HE7C3,(PEEK(&HE7C3) OR 2)AND 254"

pour sélectionner la mémoire d'écran couleur.

2° *Adresse de mémoire d'écran*

Sur MO5, la mémoire d'écran commence à l'adresse 0, sur TO7/70 en &H4000.

3° *Utilisation des couleurs*

Le TO7/70 possède 8 couleurs et le MO5 en possède 15.

4° *Implantation des programmes Assembleur*

MO5

0000 - 1F3F	: écran
1F40 - 1FFF	: libre
2000 - 20FF	: page 0 moniteur
2100 - 21FF	: page 0 BASIC
2200 - 9FFF	: libre
A000 - A7BF	: réservé pour le disque
A7C0 - A7C3	: PIA 6821
A7C4 - A7CB	: libre
A7CC - A7CF	: 6821 - Joystick
A7D0 - A7DF	: contrôleur de disque
A7E0 - A7E7	: 6821 - Interface //
A7E8 - AFFF	: libre
B000 - BFFF	: réservé cartouches
C000 - EFFF	: BASIC
F000 - FFFF	: moniteur

TO7/70

0000 - 3FFF	: réservé cartouches
4000 - 5F3F	: écran
5F40 - 5FFF	: libre
6000 - 60FF	: page 0 moniteur
6100 - 61FF	: page 0 BASIC
6200 - DFFF	: libre
E000 - E7BF	: réservé pour le disque
E7C8 - E7CB	: PIA 6821
E7CC - E7CF	: 6821 - Joystick
E7D0 - E7DF	: contrôleur de disque
E7E0 - E7E3	: 6821 - Interface //
E7E4 - E7FF	: libre (extensions)
E800 - FFFF	: moniteur

La grande place disponible de &H6200 à &HBFFF a fait choisir cet emplacement pour implanter les programmes assembleur et sauver la mémoire d'écran.

Voici les modifications à apporter aux programmes listés ci-dessus pour leur permettre de "tourner" sur TO7/70.

- *Copie d'écran monochrome :*

```
10040 A=L3*40+C3+J-C1+(I-L1)*40+&H4000
10050 B=J+I*40+&H4000
```

- *Copie d'écran multichrome :*

```
10020 POKE &HE7C3,PEEK(&HE7C3) OR 1:GOSUB 10050 'Motif
10030 POKE &HE7C3,(PEEK(&HE7C3) OR 2) AND &HFE:GOSUB
10050
10070 A=L3*40+C3+ J-C1+(I-L1)*40+&H4000
10080 B=J+I*40+&H4000
```

- *Couleur d'un caractère de l'écran :*

```
20    FOR I=1 TO 7
60    FOR I=1 TO 7
10020 POKE &HE7C3,(PEEK(&HE7C3) OR 2)AND &HFE:GOSUB
10050
10030 A=LI*40*8+CO+&H4000
10040 C=(PEEK(A)-200)/8+1
```

- *Palette de couleurs :*

```
60    FOR I=0 TO 7
2070  FOR J=0 TO 7
```

- *TAG :*

```
11015 POKE &HE7C3,PEEK(&HE7C3) OR 1
11060 POKE LI*40+C+(I-1)*40+&H4000,A
11100 POKE LI*40+C+(I-1)*40+1+&H4000,A
```

- *Repère et repère paramétrable :*

Mêmes modifications que pour TAG.

- *Sauvegarde de contexte monochrome :*

```
100 REM Sauvegarde de contexte monochrome
110 :
120 FOR I=0 TO 17
130   READ A:POKE &H9000+I,A
140 NEXT I
```

```
150 :
160 DATA &H8E,&HA0,0,&H10,&H8E,&H40,0,&HA6,&HA0,
&HA7,&H80
170 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
180 :
200 EXEC &H9000 'Sauvegarde
```

- *Lecture de contexte monochrome :*

```
100 REM Lecture de contexte monochrome
110 :
120 FOR I=0 TO 17
130   READ A:POKE &H9000+I,A
140 NEXT I
150 :
160 DATA &H8E,&HA0,0,&H10,&H8E,&H40,0,&HA6,&H80,
&HA7,&HA0
170 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
180 :
200 EXEC &H9000 'Lecture
```

- *Sauvegarde de contexte multichrome :*

```
100 REM Sauvegarde de contexte multichrome
110 :
115 POKE &HE7C3,PEEK(&HE7C3) OR 1 'Selection mot
if
120 FOR I=0 TO 17
130   READ A:POKE &H8000+I,A
140 NEXT I
150 :
160 DATA &H8E,&H90,0,&H10,&H8E,&H40,0,&HA6,&HA0,
&HA7,&H80
170 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
180 :
200 EXEC &H8000 'Sauvegarde Motif
210 :
212 POKE &HE7C3,(PEEK(&HE7C3)OR 2)AND &HFE 'Sele
ction couleur
220 FOR I=0 TO 17
230   READ A:POKE &H8000+I,A
240 NEXT I
250 :
260 DATA &H8E,&HB0,0,&H10,&H8E,&H40,0,&HA6,&HA0,
&HA7,&H80
270 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
280 :
300 EXEC &H8000 'Sauvegarde Couleur
```

- *Lecture de contexte multichrome :*

```
100 REM Lecture de contexte multichrome
110 :
115 POKE &HE7C3,PEEK(&HE7C3) OR 1 'Selection mot
if
120 FOR I=0 TO 17
130   READ A:POKE &H8000+I,A
140 NEXT I
150 :
160 DATA &H8E,&H90,0,&H10,&H8E,&H40,0,&HA6,&H80,
&HA7,&HA0
170 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
180 :
200 EXEC &H8000 'Lecture Motif
210 :
212 POKE &HE7C3,(PEEK(&HE7C3)OR 2)AND &HFE 'Sele
ction couleur
220 FOR I=0 TO 17
230   READ A:POKE &H8000+I,A
240 NEXT I
250 :
260 DATA &H8E,&HB0,0,&H10,&H8E,&H40,0,&HA6,&H80,
&HA7,&HA0
270 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
280 :
300 EXEC &H8000 'Lecture Couleur
```

- *Exemple de sauvegarde et restitution de contexte :*

```
10 REM Demonstration de sauvegarde
11 :
12 CLS
13 FOR I=1 TO 7
14   COLOR I
15   PRINT"Demonstration de sauvegarde de CTXT"
16 NEXT I
17 GOSUB 100 'Sauvegarde
18 :
19 CLS : LOCATE 0,0,0
20 PRINT"Appuyez sur une touche pour reafficher"
21 PRINT"l'ecran precedent."
22 A$=INPUT$(1)
23 GOSUB 500
24 LOCATE 0,15
25 :
26 END
27 REM ********************************
100 REM Sauvegarde de contexte multichrome
110 :
```

```
115 POKE &HE7C3,PEEK(&HE7C3) OR 1 'Selection mot
if
120 FOR I=0 TO 17
130   READ A:POKE &H8000+I,A
140 NEXT I
150 :
160 DATA &H8E,&H90,0,&H10,&H8E,&H40,0,&HA6,&HA0,
&HA7,&H80
170 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
180 :
200 EXEC &H8000 'Sauvegarde Motif
210 :
212 POKE &HE7C3,(PEEK(&HE7C3)OR 2)AND &HFE 'Sele
ction couleur
220 FOR I=0 TO 17
230   READ A:POKE &H8000+I,A
240 NEXT I
250 :
260 DATA &H8E,&HB0,0,&H10,&H8E,&H40,0,&HA6,&HA0,
&HA7,&H80
270 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
280 :
300 EXEC &H8000 'Sauvegarde Couleur
310 :
320 RETURN
330 REM *****************************
500 REM Lecture de contexte multichrome
510 :
515 POKE &HE7C3,PEEK(&HE7C3) OR 1 'Selection mot
if
520 FOR I=0 TO 17
530   READ A:POKE &H8000+I,A
540 NEXT I
550 :
560 DATA &H8E,&H90,0,&H10,&H8E,&H40,0,&HA6,&H80,
&HA7,&HA0
570 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
580 :
600 EXEC &H8000 'Lecture Motif
610 :
612 POKE &HE7C3,(PEEK(&HE7C3) OR 2)AND &HFE 'Sel
ection couleur
620 FOR I=0 TO 17
630   READ A:POKE &H8000+I,A
640 NEXT I
650 :
660 DATA &H8E,&HB0,0,&H10,&H8E,&H40,0,&HA6,&H80,
&HA7,&HA0
670 DATA &H10,&H8C,&H5F,&H40,&H26,&HF6,&H39
```

```
680 :
700 EXEC &H8000 'Lecture Couleur
710 :
720 RETURN
```

- *Recherche d'un octet en mémoire :*

```
10000 REM ***********************************
10001 REM *                                  *
10002 REM *  RECHERCHE D'UN OCTET EN MEMOIRE   *
10003 REM *                                  *
10004 REM ***********************************
10005 REM *                                  *
10006 REM * Entree  : AD=Adresse de depart    *
10007 REM *           O =Octet recherche      *
10008 REM * Sortie  : A =1ere Adresse contenant A*
10009 REM *                                  *
10010 REM ***********************************
10011 :
10020 MSB=INT(AD/256):LSB=AD-MSB*256 'Calcul Poid
s faible et fort Adresse
10030 POKE &H8F42,MSB:POKE&H8F43,LSB 'Memorisatio
n Adresse
10040 POKE &H8F40,O 'Memorisation Octet recherche
10045 :
10050 FOR I=0 TO 13
10060   READ A
10070   POKE &H8000+I,A
10080 NEXT I
10100 DATA &HBE,&H8F,&H42,&HA6,&H80,&HB1,&H8F,&H4
0,&H26,&HF9,&HBF,&H8F,&H44,&H39
10105 :
10110 EXEC&H8000 'Recherche
10120 A=PEEK(&H8F44)*256+PEEK(&H8F45)-1
10130 :
10140 RETURN
```

- *Recherche de deux octets en mémoire :*

```
10000 REM ***********************************
10001 REM *                                  *
10002 REM *  RECHERCHE DE 2 OCTETS EN MEMOIRE  *
10003 REM *                                  *
10004 REM ***********************************
10005 REM *                                  *
10006 REM * Entree  : AD=Adresse de depart    *
10007 REM *           O1=1er  octet recherche *
10008 REM *           O2=2eme octet recherche *
10009 REM * Sortie  : A =1ere Adresse contenant A*
10010 REM *                                  *
10011 REM ***********************************
```

```
10012 :
10020 MSB=INT(AD/256):LSB=AD-MSB*256 'Calcul poid
s faible et fort Adresse
10030 POKE &H8F42,MSB:POKE&H8F43,LSB
10040 POKE &H8F40,01:POKE&H8F41,02
10045 :
10050 FOR I=0 TO 20
10060   READ A
10070   POKE &H8000+I,A
10080 NEXT I
10100 DATA &HBE,&H8F,&H42,&HA6,&H80,&HB1,&H8F,&H4
0,&H26,&HF9,&HA6,&H80
10101 DATA &HB1,&H8F,&H41,&H26,&HF4,&HBF,&H8F,&H4
4,&H39
10105 :
10110 EXEC&H8000 'Recherche
10120 A=PEEK(&H8F44)*256+PEEK(&H8F45)-2
10130 :
10140 RETURN
```

Conclusion

Pour conclure ce manuel, précisons deux points :

□ Nous venons de voir, par de nombreux exemples, comment créer des modules utilitaires afin d'augmenter les possibilités de MO5 ou TO7/70.

Cette liste d'utilitaires n'est pas exhaustive ; si vous éprouvez le besoin de créer une nouvelle fonction sur votre MO5 ou TO7/70, la méthode développée dans ce livre peut être fructueuse :

- définissez les entrées/sorties du module,
- structurez et hiérarchisez son traitement,

et vous verrez que la tâche est facilitée.

□ Il serait faux et prétentieux de dire que ces programmes s'adaptent sans difficulté à des micro-ordinateurs différents des MO5 et TO7/70. En particulier, les techniques développées dans les chapitres 4 et 5 (le générateur sonore et les modes graphiques haute résolution) sont très spécifiques à MO5 et TO7/70 et difficilement adaptables à une autre machine.

Par contre, tout le chapitre 2 (les améliorations du BASIC MO5 et TO7/70) qui développe des utilitaires est facilement adaptable.

Alors, si vous possédez un autre micro-ordinateur que MO5 ou TO7/70, et si vous avez ... un peu de courage, je vous souhaite bonne chance !

Index

Conseils de lecture

Pour approfondir vos connaissances en BASIC Thomson et mieux connaître le système du MO5 et du TO7/70, P.S.I. vous propose une palette d'ouvrages utiles.

POUR MAITRISER LE BASIC DES MO5 ET TO7/70

□ **MO5 et TO7/70, Méthodes pratiques** – Jacques Boisgontier (Éditions du P.S.I.)

Pour ceux qui ont déjà pratiqué un BASIC, un ouvrage de perfectionnement du BASIC Thomson, illustré par de nombreux programmes-exemples.

□ **Le BASIC des MO5 et TO7/70** – Gilles Blanchard (Éditions du P.S.I.)

Une initiation pratique au BASIC des MO5 et TO7/70 précisant les différences existantes entres ces deux machines.

POUR VOUS INITIER AU LANGAGE MACHINE THOMSON

□ **Assembleur et périphériques des MO5 et TO7/70** – Frédéric Blanc et François Normant (Éditions du P.S.I.)

Une initiation progressive et claire à l'assembleur 6809 et à la programmation en langage machine de tous les périphériques disponibles sur MO5 et TO7/70.

□ **Clefs pour MO5** – Gilles Blanchard (Éditions du P.S.I.)

Mémento présentant synthétiquement le jeu d'instructions du 6809, le PIA système, les adresses utiles pour exploiter les périphériques du MO5 ; le livre de chevet du programmeur sur MO5.

□ **Clefs pour TO7/70** – Gilles Blanchard (Éditions du P.S.I.)

Le pendant du *Clefs pour MO5* pour les possesseurs de TO7/70.

POUR ÊTRE INFORMÉ RÉGULIÈREMENT DE L'ACTUALITÉ DES MICROS THOMSON

□ **MICROTOM,** revue bimestrielle du Groupe Tests.

Pour exploiter au mieux les capacités de votre micro, vous trouverez au sommaire de chaque numéro, un rendez-vous avec les rubriques clés :

– Apprivoisez votre Thomson : idées, astuces, conseils, toutes les recettes pour comprendre votre ordinateur, son anatomie, son fonctionnement, sa programmation et exploiter ses capacités graphiques et sonores.

– Explorez l'univers de votre MO5-TO7 TO7/70 : logiciels, périphériques, langages, toute l'actualité des ordinateurs Thomson au rendez-vous de MICROTOM.

– Programmez votre micro Thomson : hobbystes, enseignants, petits et grands, fanatiques ou même encore débutants, dans chaque numéro de MICROTOM une palette de programmes de tous niveaux (jeux utilitaires, pédagogiques, etc.).

Votre avis nous intéresse

- Pour nous permettre de faire de meilleurs livres, adressez-nous vos critiques sur le présent livre.
- Si vous souhaitez des éclaircissements techniques, écrivez-nous, nous adresserons votre demande à l'auteur qui ne manquera pas de vous répondre directement.

- Ce livre vous donne-t-il toute satisfaction?

____
____
____
____
____

- Y a-t-il un aspect du problème que vous auriez aimé voir abordé?

____
____
____
____
____

Comment avez-vous eu connaissance de ce livre?

☐ publicité
☐ catalogue
☐ boutique micro
☐ autres
☐ cadeau
☐ librairie
☐ exposition

Avez-vous déjà acquis des livres PSI?

lesquels? ______________________________
____

qu'en pensez-vous? ______________________________
____
____

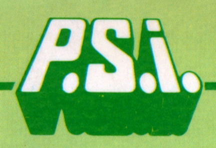
Nom ______________ Prénom ______________ Age ______
Adresse ______________________________
Profession ______________________________
Centre d'intérêt ______________________________

CATALOGUE GRATUIT

Vous pouvez obtenir un catalogue complet des ouvrages PSI, sur simple demande, ou en retournant cette page remplie à votre libraire, à votre boutique micro ou aux

Editions du PSI
BP 86
77402 Lagny-sur-Marne Cedex

Achevé d'imprimer par l'imprimerie Tardy Quercy S.A. à Bourges
D. L. : Octobre 1985 - N° d'Imprimeur : 12570 - N° d'Édition : 86595-264-1



BASIC PLUS

80 ROUTINES SUR
MO5 ET TO7/70

Vous connaissez bien le BASIC du MO5 ou du TO7/70, et, sans songer encore à l'assembleur, vous souhaitez accroître les capacités de votre Thomson : "BASIC plus" vous propose 80 routines pour "muscler" votre ordinateur, 80 manières de simuler des fonctions que vous n'auriez jamais cru pouvoir utiliser.

"**B**asic plus" vous permet de réaliser des copies d'écran en basse résolution et vous dévoile les possibilités du synthétiseur de son pour programmer un morceau de musique ou pour faire un métronome de votre Thomson.

"**M**ode graphique haute résolution" simule deux ordres graphiques évolués : CIRCLE et ARC.

Un dernier chapitre sur l'animation graphique vous indique comment créer des caractères graphiques et constituer des images-écran avec le crayon optique.

Au-delà du BASIC MO5 et TO7/70, découvrez BASIC plus !



ÉDITIONS DU P.S.I.
BP 86 – 77402 LAGNY S/MARNE CEDEX-FRANCE

ISBN : 2 86595-264 9 **105 F.F.**

PHOTO : LAURENT MAUNOURY

BASIC PLUS 80 ROUTINES SUR **MO5 ET TO7/70**

P.S.I.